WRDC-TR-89-3045

AEROSPACE STRUCTURES DESIGN ON COMPUTERS

Vipperla B. Venkayya

Analysis and Optimization Branch
Structures Division

Final Report for Period December 1988 - March 1989

March 1989

Approved for public release; distribution is unlimited.

FLIGHT DYNAMICS LABORATORY
WRIGHT RESEARCH and DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6523

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

VIPPERLA B. VENKAYYA
Aerospace Engineer
Design & Analysis Methods Group

NELSON D. WOLF, Technical Manager
Design & Analysis Methods Group
Analysis & Optimization Branch

FOR THE COMMANDER

JOHN T. ACH, Chief
Analysis & Optimization Branch
Structures Division

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WRDC/FIBRA, WPAFB, OH 45433-6523 to help us maintain a current mailing list

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release, Distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) WRDC-TR-89-3045	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Analysis & Optimization Branch Structures Division 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) WRDC/FIBRA Wright-Patterson AFB OH 45433-6553	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Flight Dynamics Laboratory 8b OFFICE SYMBOL (If applicable) WRDC/FIBRA	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N/A
8c. ADDRESS (City, State, and ZIP Code) Wright-Patterson AFB OH 45433-6553	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 62201F; PROJECT NO. 2401; TASK NO. 02; WORK UNIT ACCESSION NO. 76

11. TITLE (Include Security Classification)
Aerospace Structures Design on Computers

12. PERSONAL AUTHOR(S)
Vipperla B. Venkayya

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM Dec 88 TO Mar 89	1989, March	217

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	
12	01		

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This report, prepared for training, is intended to bring out the elements of structural design optimization on modern computers. The first section gives a cursory description of the requirements and essential disciplines involved in aircraft structural design. The second section is an optimization paper that provides the basis for optimization using large finite element assemblies. The third section provides a summary of design sensitivity analysis which is an essential element of optimization. The two appendices are the descriptions of two training programs for analysis and optimization. Each of these sections has their own references. This is an informal report itended for training and is a collection of material entirely from the open literature.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL
Vipperla B. Venkayya	513-255-7191	WRDC/FIBR

DD Form 1473, JUN 86 *Previous editions are obsolete.* SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

FOREWORD

The purpose of this technical report is to provide a cursory outline of structural optimization. It is an informal report, intended for training. The material is collected entirely from the open literature.

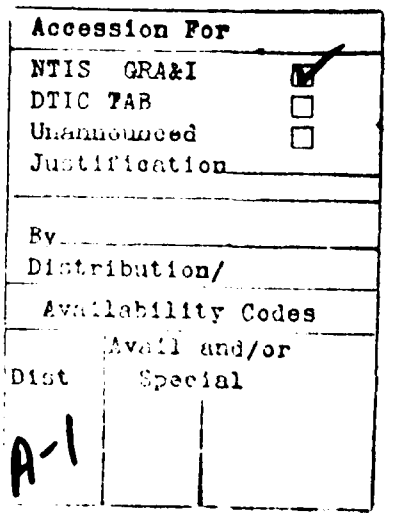


TABLE OF CONTENTS

SECTION	TITLE	PAGE
1.0	INTRODUCTION	1
2.0	REQUIREMENTS FOR AIRCRAFT STRUCTURAL DESIGN	3
3.0	OPTIMIZATION PAPER	35
4.0	DESIGN SENSITIVITY ANALYSIS	65
APPENDIX		
A	OPTSTAT REPORT	90
B	LISTING OF THE PROGRAM	

1.0 INTRODUCTION

In modern times more and more tasks of engineering design are being relegated to computers because of their immense computing power and versatility. The new computers offer significant opportunities for advancing computer-aided design in the true sense. Design of a total system with all the complexities of the interacting disciplines may be a reality in the not too distant future. Integrated engineering optimization systems are in development around the world in pursuit of this goal. The implications of this scenario are far reaching in improving product quality and reliability while reducing cost and design time.

The flip side of this scenario is concern about mindless automation and its implications on creativity. It is disconcerting to see young engineers spending all their productive time in front of computer terminals believing results from the black box with little concern or understanding of the modeling nuances and errors. The most frequently asked question is: Is design automation really reducing manpower and time or simply creating a quagmire? Are we really designing more airplanes in a shorter time than in the 50s and 60s? The answer is probably negative. However, there is no question that modern systems are more complex and performance goals are much more stringent, and they cannot be met without extensive trade off studies and optimization on supercomputers. A thorough understanding of the disciplines and the design requirements is as important now as before. Reliance on ready made design software (black boxes) without this understanding is counter productive.

This report, prepared for training, is intended to bring out the elements of structural design optimization on modern computers. The first section gives a cursory description of the requirements and essential disciplines involved in aircraft structural design. The second section is an optimization paper that provides the basis for optimization using large finite

element assemblies. The third section provides a summary of design sensitivity analysis which is an essential element of optimization. The two appendices are the descriptions of two training programs for analysis and optimization. Each of these sections has their own references. This is an informal memo intended for training and is a collection of material entirely from the open literature.

2.0 REQUIREMENTS FOR AIRCRAFT STRUCTURAL DESIGN

The structural design requirements of an aircraft are derived from a number of disciplines. Aircraft design is generally a group effort and effective communication between the groups is essential for designing optimum structures as well as to reduce design time and cost. This effective communication can be established if each group has at least a rudimentary understanding of the functions of the other groups. This interdisciplinary communication is becoming even more important as the design functions are delegated more and more to computers. The interaction between the following groups is very much desirable in structural optimization.

1. Loads (Aerodynamics, Ground Loads, etc.)

2. Structures

3. Weight and Balance/Mass Properties

4. Power Plant Analysis

5. Materials and Processes

6. Controls Analysis

Loads

Like all other structures the aircraft must be designed to withstand the loads induced by the environment in which it operates. The loads on the aircraft can be classified into three broad categories:

1. Maneuver Loads

2. Ground Loads

3. Turbulence

Maneuver Loads: Air Loads & Inertia Loads

The maneuver loads are generally air loads resulting from the way the aircraft operates. These maneuvers can be classified into the following simple movements of the aircraft.

1. Forward Acceleration

2. Roll

3. Pitch

4. Yaw

5. Pitch and Yaw

6. Roll and Pitch

7. Roll and Yaw

8. Roll, Pitch and Yaw

The first three maneuvers will have the angle of yaw zero and no yawing couple, and they are regarded as symmetrical maneuvers. In all the others the angle of yaw and the yawing couple will not both be zero and these are termed asymmetrical maneuvers. The forces applied to the aircraft are the aerodynamic forces on the external surfaces, the gravitational forces, and the forces from the propulsion unit. These forces are governed by

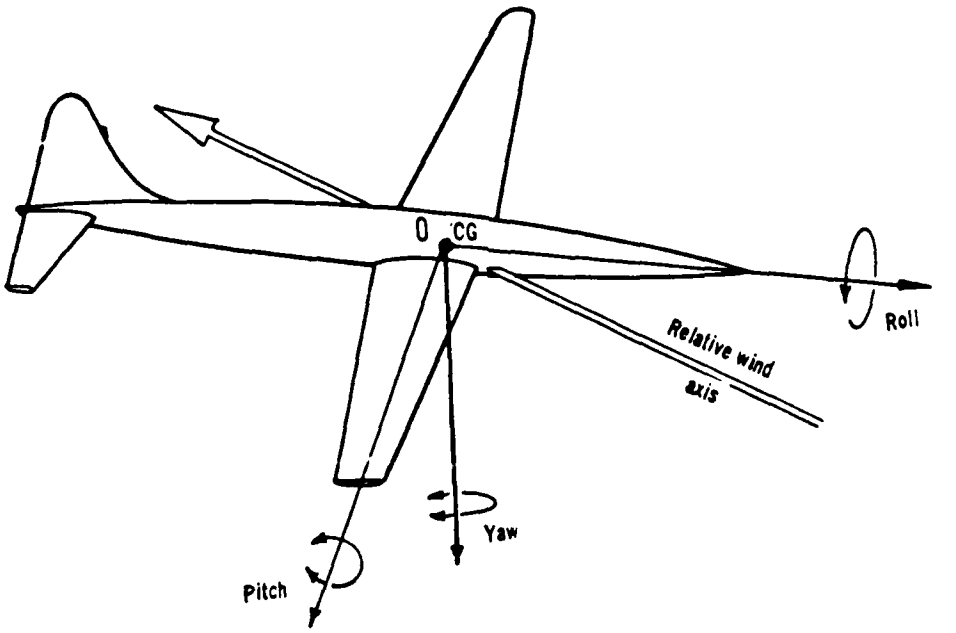


Fig 1: Simple Movements of the Aircraft

Newton's laws of motion and they can be derived from basic momentum equations. The equations of motion relative to the principal axes of inertia can be written as

$$X = m(\dot{U} - rV + qW) \tag{1}$$

$$Y = m(\dot{V} - pW + rU) \tag{2}$$

$$Z = m(\dot{W} - qU + pV) \tag{3}$$

$$L = A\dot{p} + (C - B)qr \tag{4}$$

$$M = B\dot{q} + (A - C)rp \tag{5}$$

$$N = C\dot{r} + (B - A)pq \tag{6}$$

The aircraft's principal inertia axes are shown in Figure 2. X, Y, Z are the forces in the directions X, Y, Z. m is the total mass of the aircraft. L, M, N are the moments about the axes X, Y, Z respectively. A, B, C are the moments of inertia of the aircraft about the same axes. U, V, W are the velocities (translational) and p, q, r are the angular velocities in the direction and about the principal axes.

For small angles of rotation the equations of motion can be linearized and simplified. For simple maneuvers listed earlier the linearized equations can be written as follows:

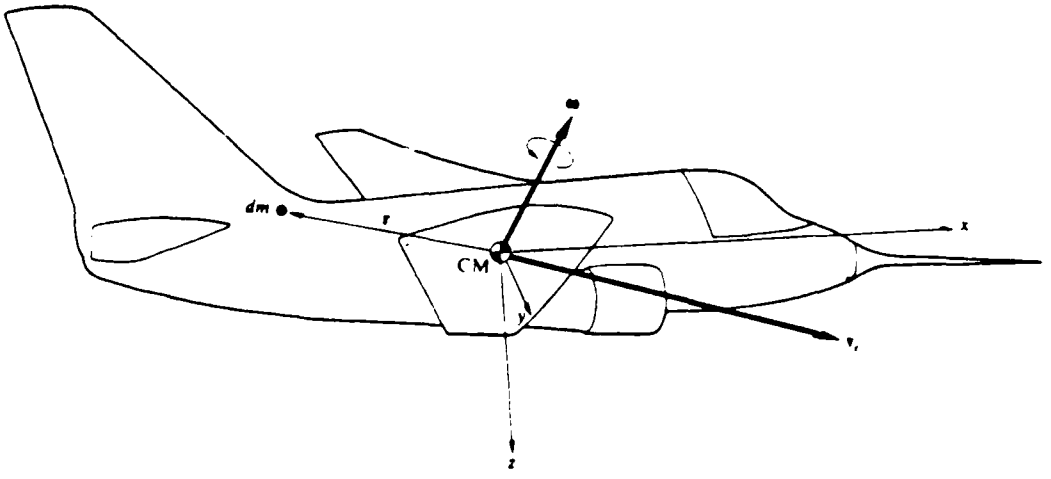


Fig 2: Aircraft's Principal Inertia Axes

1. Forward Acceleration

$$X = m\dot{U} \tag{7}$$

2. Pure Roll (under very restrictive conditions)

$$L = A\dot{p} \tag{8}$$

3. Pure Pitch

$$Z = m(\dot{W} - qU) \qquad M = B\dot{q} \tag{9}$$

4. Pure Yaw

$$Y = m(\dot{V} + rU) \qquad N = C\dot{r} \tag{10}$$

5. Pitch and Yaw

$$Y = m(\dot{V} + rU) \qquad Z = M(\dot{W} - qU) \qquad M = B\dot{q} \qquad N = C\dot{r} \tag{11}$$

For the other maneuvers all six equations (1-6) are involved. For any of these maneuvers to be attainable it must be possible to apply the three control couples separately and the trim of the aircraft in the other directions to be unaltered.

In all of the equations listed so far the left-hand side represents the applied force or couple at the C-G of the aircraft, and the right-hand side represents the rate of change of momentum or moment of momentum. The aero dynamic forces, the engine thrust and the inertia forces provide the left-hand side. They depend on the distortion and displacement of the whole aircraft relative to the direction of flight under the action of the controls. The force-moment equations written so far describe the gross movement of the aircraft and they are referred to the motion of the C-G of the aircraft. However, for the design

of an aircraft we need to determine the distribution of the aerodynamic forces (in the form of lift forces) on the external surfaces. For example we need to know the chordwise and spanwise distribution of the aerodynamic forces on the lifting surfaces like the wing, horizontal stabilizer and the fin.

The pressure distribution on the lifting surfaces can be expressed as

$$\underset{\sim}{P} = \underset{\sim}{A}\underset{\sim}{W} \tag{12}$$

where $\underset{\sim}{P}$ is the resultant pressure on each panel. It is assumed that the lifting surface is divided into a number of panels. The sides of the panel are assumed to be parallel to the free stream (See Figure 3) and the pressure is assumed to be constant over each panel. $\underset{\sim}{A}$ is the aerodynamic influence coefficient matrix the elements of which can be calculated by aerodynamic theories such as vortex-lattice or doublet lattice for the subsonic cases and supersonic distribution or mach box theory for the supersonic cases. The matrix $\underset{\sim}{W}$ represents the downwash distributions which generally consist of rigid surface inclinations to the free stream and deflections of the control surfaces. The rigid surface inclinations include the effective angle of attack of the surface, local incremental angles of attack due to camber and twist and additive corrections to the local incidences. The effective angle of attack equals the sum of the geometric angle of attack of the wing relative to the fuselage, the inclination of the fuselage, and the upwash induced by this inclination.

Mass Properties: Inertia Loads

In addition to the aerodynamic forces, each maneuver is associated with inertia loads. These inertia loads are either due to gravity or any maneuver involving acceleration of the aircraft. To calculate the inertia loads we need to know, at least approximately, the

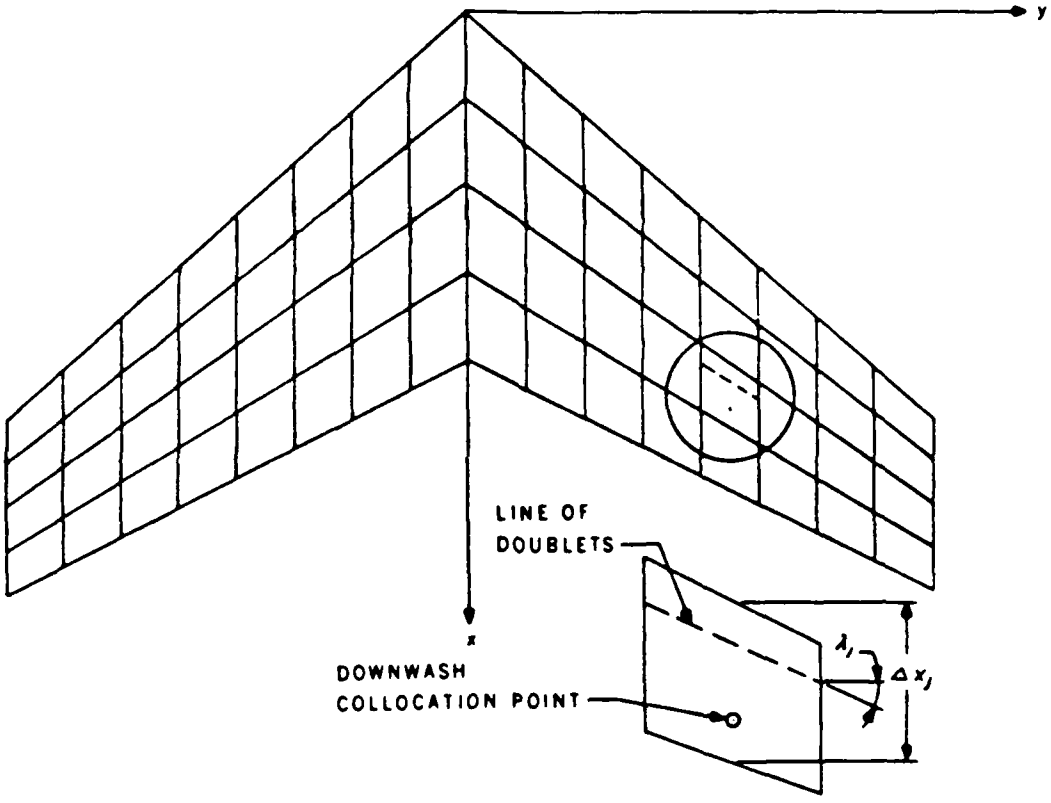


Fig 3: Idealization of a Wing Panel into Boxes

mass properties of the aircraft. The total mass of the aircraft is made up of structural and non-structural parts. The analytical models can only estimate the structural mass of the aircraft. The non-structural mass properties are generally estimated from the past experience of similar aircraft. These estimates have to be continuously revised as the detailed design of the aircraft evolves. Once the mass properties are known the inertia forces can be estimated by application of Newton's second law of motion.

Aerodynamic Surfaces - Structural Boxes

In most aircraft lifting surfaces the structural box is only a fraction of the total and the rest of it is made up of control surfaces and surfaces to enhance the lift area. The structural boxes are generally approximated by finite element grids, while the entire lifting surface is divided into aerodynamic panels for the purpose of calculating the pressure distributions. The total panel loads can be calculated and the center of pressure points can be determined. However, these load points and the structural grids do not generally coincide. For structural analysis these loads have to be transformed from the aerodynamic grid to the structural grid. These transformations can be carried out by polynomial or spline interpolations. A similar situation arises when we are considering aeroelastic effects (flexibility effects) on the airload distribution. Here the structural box deformations have to be extrapolated to obtain the correct angle of attack. The same polynomial or spline extrapolation can be used.

Ground Loads

The ground loads are a result of three distinct conditions:

(i) Taxying

(ii) Take-off

(iii) Landing

The runway profile and the time spent taxying at different speeds are the important factors contributing to the taxy loads. The discrete bumps or chuck holes can significantly increase the taxy loads. The aircraft flexibility also significantly effects this load.

In most cases the take-off may be considered an extension of the taxying condition. The conditions governing the landing loads are distinctly different from any of the other two. The attitude of the aircraft and the resulting ground loads can be fully defined if the following parameters are known:

(i) Vertical Velocity at Touch Down

(ii) Horizontal Velocity

(iii) Bank Angle

(iv) Rolling Angular Velocity

(v) Yaw Angle

(vi) Yawing Angular Velocity

(vii) Pitch Angle

(viii) Pitching Angular Velocity

The actual distribution of the ground loads to various components of the aircraft cannot be quite precise but empirical estimates would be adequate.

Material Properties - Strength

In order to correctly define the strength constraints (strength margins of safety) we must clearly understand the material properties of the structure. The material strength in the allowable properties of the material are based on these factors:

* Allowable stresses based on yield or ultimate strength.

* Allowable stresses based on local buckling or crippling.

* Allowable properties based on durability and damage tolerance.

The yield or ultimate strength of the material is simply a metallurgical property, and they are determined by simple tensile (or compression) coupon (uniaxial) tests or torsion beam tests.

The local buckling or crippling strength depends on the material property as well as the geometry of construction of the structural elements. Simple example are column buckling, local panel buckling, stiffener buckling, beam buckling, etc.

The durability and damage tolerance considerations are much more involved. Fatigue life and fracture mechanics considerations are of extreme importance in aircraft design. In defining strength constraints we must take full cognizance of the fatigue and fracture properties of the materials.

Allowable Stresses Based on Yield/Ultimate Strength

The material allowable strength is generally determined from uniaxial coupon or torsion beam tests. In a uniaxial state of stress the stress in the element can be limited to its tension or compression allowable. Usually the allowable stress is specified as some fraction of the tensile or compressive yield strength. This fraction depends on the desired factor of safety. In some materials the stress allowable may not be the way to specify the material constraint. In such cases the strain allowable may be more appropriate. Similarly in

the case of elements predominately subjected to shear, an allowable shear stress can be specified.

Most structural elements are (in particular, surface elements) in a biaxial state of stress. In such cases a failure theory has to be invoked to specify a stress constraint based on material strength. The most commonly used failure theories for metals in a biaxial state of stress are:

1. Energy of Distortion or Von Mises Criterion.

2. Tresca's Shear Stress Criteria.

Both theories give comparable results and for our present discussion we will adopt the energy of distortion theory. In most general terms the modified energy of distortion theory can be stated as follows:

$$\sqrt{\left(\frac{\sigma_x}{X}\right)^2 + \left(\frac{\sigma_y}{Y}\right)^2 - \frac{\sigma_x \sigma_y}{XY} + \left(\frac{\sigma_{xy}}{Z}\right)^2} \leq 1 \tag{13}$$

where σ_x, σ_y, σ_{xy} represent the actual stress state in the element's local reference axis. X, Y and Z are the allowable stresses in the respective directions. The tension and compression allowables can be different, in which case there are five allowable stresses for each material. For some materials uniaxial strain allowables may be more appropriate. For the case of solid elements in a state of three dimensional stress, an octahedral shear stress criteria would be more appropriate. However, three dimensional elements are not relevant for the present discussion of optimization.

In many aircraft specifications the stress constraints in the elements are specified in terms of margins of safety (MS) which can be defined as

$$\text{MS} = \frac{1 - \text{ESR}}{\text{ESR}} \tag{14}$$

where ESR, the effective stress-ratio, is defined as

$$\text{ESR} = \sqrt{\left(\frac{\sigma_x}{X}\right)^2 + \left(\frac{\sigma_y}{Y}\right)^2 - \frac{\sigma_x \sigma_y}{XY} + \left(\frac{\sigma_{xy}}{Z}\right)^2} \tag{15}$$

Generally a specified positive margin of safety (MS) is required in most aircraft design.

Allowable Stresses Based on Local Buckling

Most aircraft elements are light and flimsy because of the overriding requirements of structural weight reduction to increase the payload and reduce the fuel consumption. Local buckling is a potential failure mode and it can occur substantially below the material strength. In such cases the allowable stresses for the elements must be determined by buckling considerations. These buckling stresses can be calculated by the following formulas:

Column Buckling

$$\sigma_{cr} = k_c \frac{E}{(L/r)^2} \tag{16}$$

Plate Buckling in Simple Compression or Shear

$$\sigma_{cr} = k_p \frac{E}{(b/t)^2} \tag{17}$$

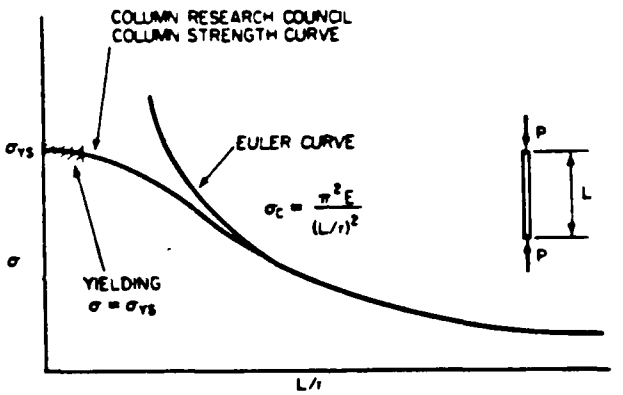


Fig 4: Column Instability

Beam Buckling: Lateral Torsional Buckling

$$\sigma_{cr} = k_B \frac{E}{\frac{Ld}{bt}} \tag{18}$$

where k_c, k_p and k_B represent the buckling constants which are functions of the element boundary conditions and loading. E is the modulus of elasticity of the material. The quantities (L/i), (b/t) and (Ld/bt) represent the slenderness ratios of the elements. Since the present optimization discussion is limited to elastic cases, we will not address buckling in the inelastic region.

Allowable Properties Based on durability and/or Damage Tolerance

Fatigue and fracture mechanics are the driving factors in this case. Every structural component is subjected to cyclic loads in service, and the fatigue properties of the design must be evaluated for adequacy. In the context of optimization the stress constraints definition must take full cognizance of the fatigue life requirements. The cyclic load on a structural component can be described by two of the six terms relating to the stress cycle.

$$S_{max} = \text{Maximum Stress}$$

$$S_{min} = \text{Minimum Stress}$$

$$S_m = \text{Mean Stress} = \frac{S_{max} + S_{min}}{2}$$

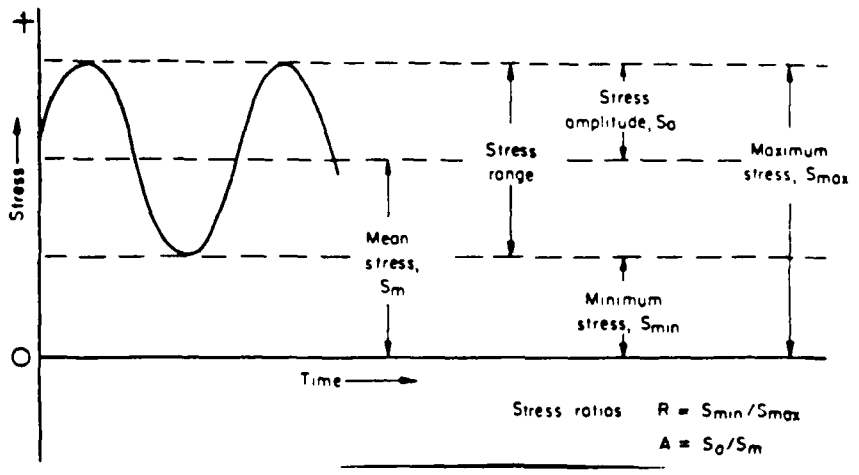
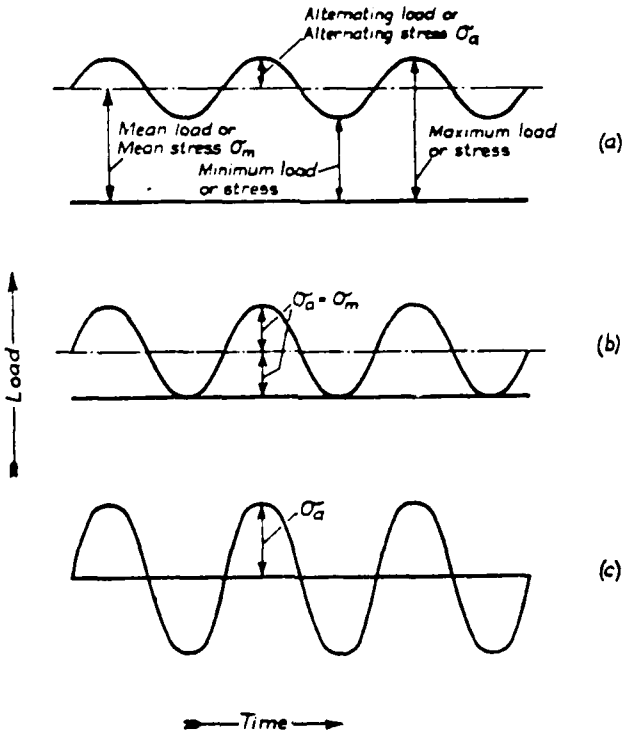


Fig 5: Nomenclature for Conventional Laboratory Fatigue Testing



(*a*) Fluctuating tension load cycle;
(*b*) Repeated load cycle;
(*c*) Reversed load cycle.

Fig 6: Types of Load Cycles

$$S_a = \text{Stress Amplitude} = \frac{S_{max} - S_{min}}{2}$$

$$S_r = \text{Stress Range}$$

$$R = \text{Stress Ratio} = \frac{S_{min}}{S_{max}}$$

The value of the stress ratio for the fully reversed stress cycle is -1, and most S-N curves for metals are given for this case.

To assess whether or not the nominal cyclic stress state will result in failure in a given number of cycles, the stress state is compared to the three criteria of failure:

1. Crack Initiation

2. Crack Propagation

3. Gross Yielding

If the stress state in question is equal to or greater than the allowable stress for crack initiation, a fatigue crack will develop in a relatively few cycles. If the stress state is equal to or greater than the allowable stress for crack propagation, any crack already present or which develops because the crack initiation criterion has been exceeded, will propagate to failure in less than the desired life. The gross-yield criteria postulates that if a nominal stress state is equal to or greater than the yield strength of the material, that stress state should be considered unsafe for long life applications.

The crack initiation for a uniaxial state of stress can be written as

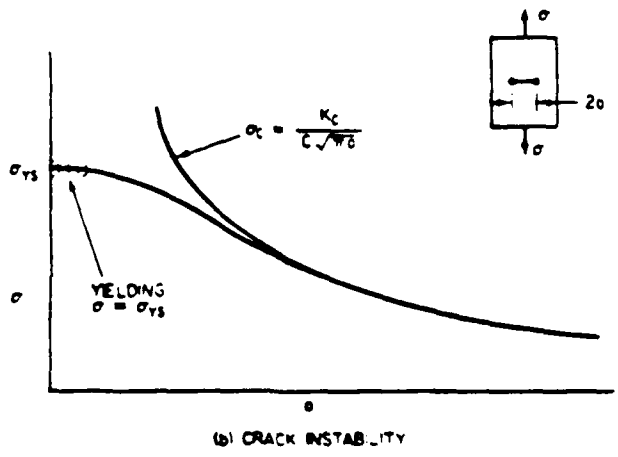


Fig 7: Column Instability and Crack Instability

$$S_a + \frac{m}{\sqrt{2}} S_m \geq \frac{S_N}{K_f} \tag{19}$$

where

S_N = Axially loaded fatigue strength at the desired life.

m = Influence of the mean stress on the allowable alternating stress.

K_f = Fatigue Notch Factor.

The modifying factor m depends on the material. A value of $m = 0.5$ is reasonable for most metals. The actual value for an aluminum alloy is $m = 0.425$. An accurate value of m may be determined from experimental data.

The criteria for crack propagation is based on the alternating tensile stress. Fatigue cracks will propagate if the alternating tensile stress is equal to or greater than the critical alternating tensile stress for propagation:

$$S_{ta} > S_{pc}$$

where S_{ta} is given by

$$S_{ta} = (S_{\text{max tensile}} - S_{\text{min tensile}})/2$$

S_{pc} = Critical alternating tensile stress to propagate a crack.

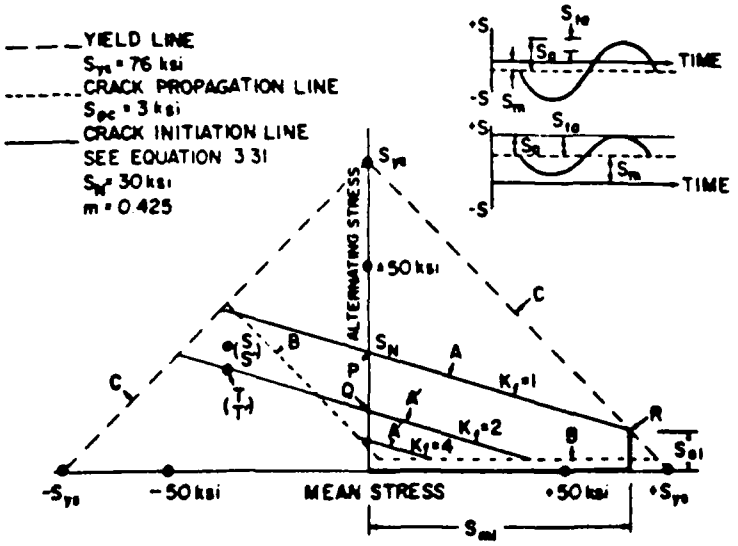


Fig 8: Theories of Failure for Unidirectional Stress (7075-T6 alloy)

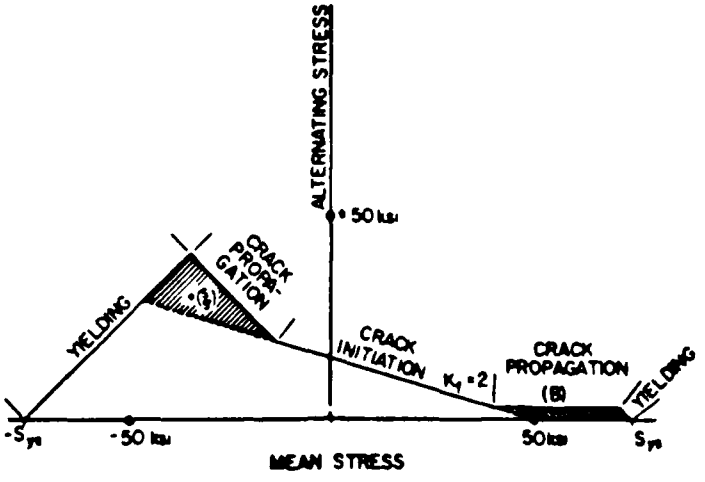


Fig 9: Example of Failure Diagram, Unidirectional Stress (7075-T6 Alloy)

The yield criterion states

$$S_a + S_m \geq S_{ys}$$

where S_{ys} = uniaxial tensile yield strength

In summary, then, in order to use the three criteria of failure to assess a given nominal stress condition, the following information must be known:

1. Fully reversed, axially loaded fatigue limit or fatigue strength for the desired number of cycles, SN.

2. Coefficient of the influence of the mean stress on the allowable alternating stress, m.

3. Critical alternating tensile stress to propagate a crack, S_{pc}.

4. Uniaxial tensile yield strength, Sys.

5. Fatigue Notch Factor, K_f, for fully reversed loading without residual stress.

6. Residual Stress State.

Additional information can be obtained from constant life fatigue diagrams or Goodman diagrams. Some examples are given in Figure 10.

Fracture Mechanics Considerations:

The damage tolerance properties of the structural elements must be determined from fracture mechanics considerations. Most built up structures will have flaws either at the joints or even at the interior of the elements due to improper finish of the components. These flaws can precipitate below the yield strength failures. In defining stress constraints

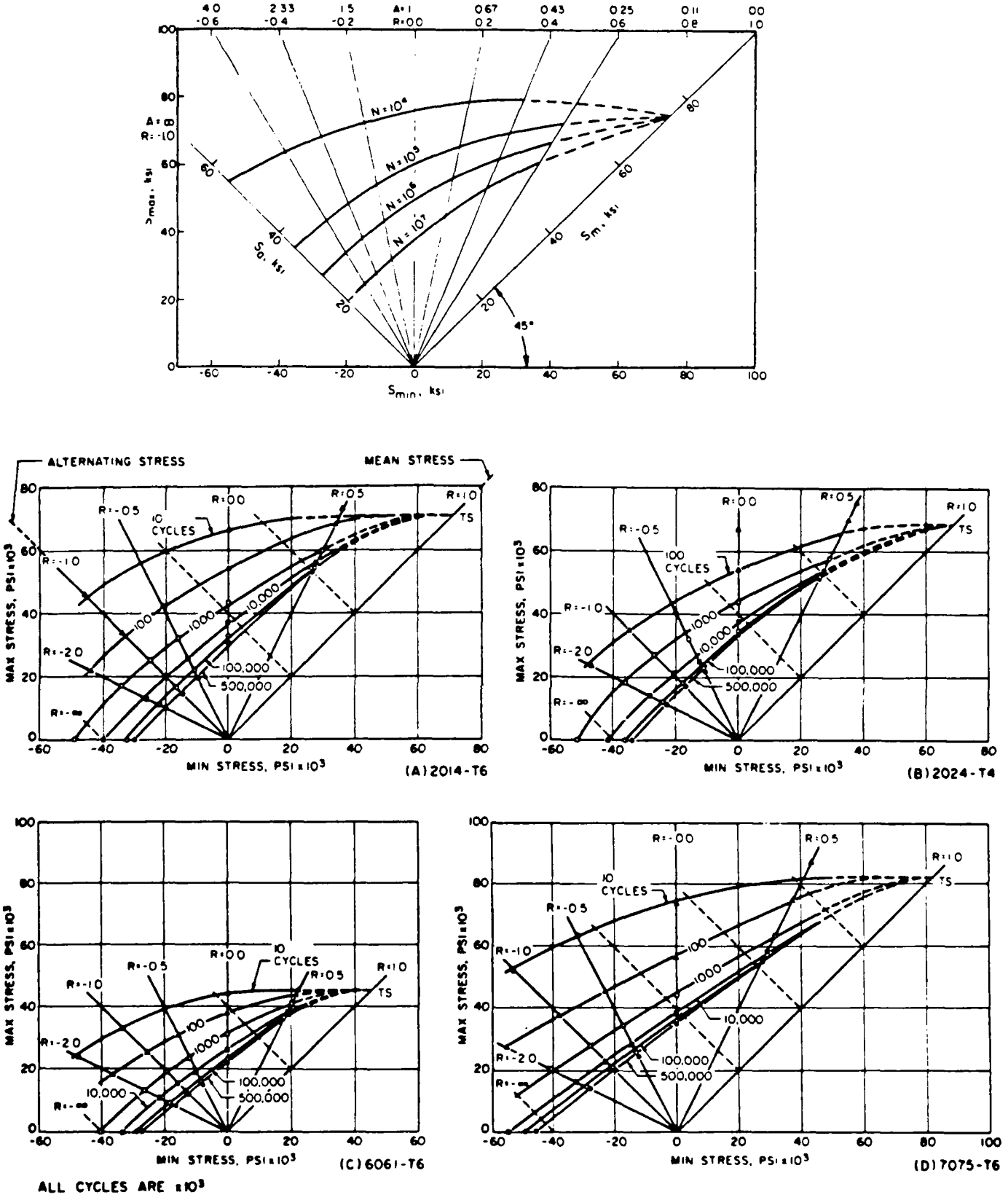


Fig 10: Constant Life Fatigue Diagrams for Several Structural Aluminum Alloys

one should be cognizant of fracture considerations. The fracture mechanics considerations are supposed to answer the following questions:

a) What is the residual strength as a function of the crack size?

b) What size crack can be tolerated at the expected service load, i.e. what is the critical crack size?

c) How long does it take for a crack to grow from a certain initial size to the critical size?

d) What size of pre-existing flaws can be permitted at the moment the structure starts its service life?

e) How often should the structure be inspected for cracks?

For our purpose we will briefly discuss the concepts of stress intensity factor and fracture toughness properties. Consider a plate with an elliptical hole

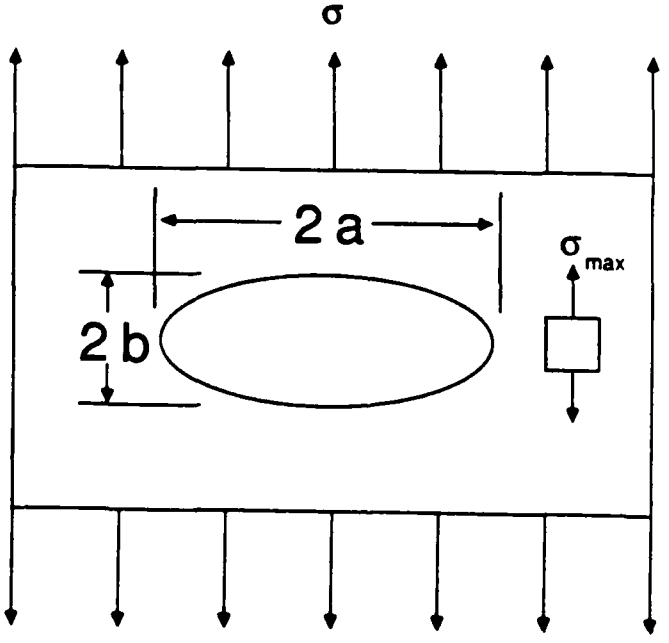


Fig. 11 Plate with an Elliptical Hole

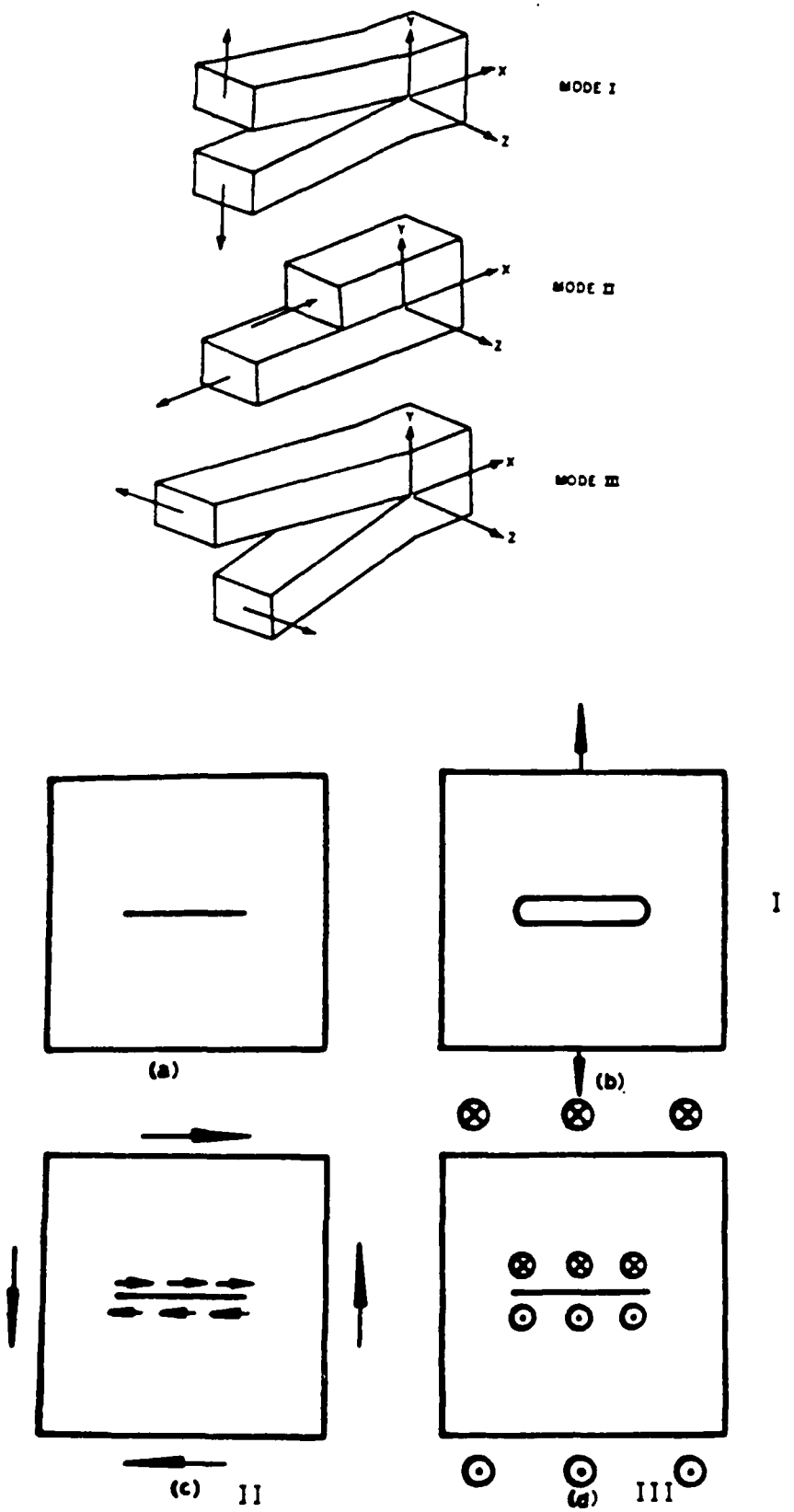


Fig 12: The Three Basic Modes of Crack Surface Displacements

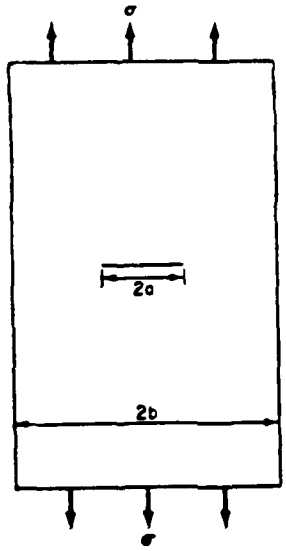


Fig 13: Finite-Width Plate Containing a Through-Thickness Crack

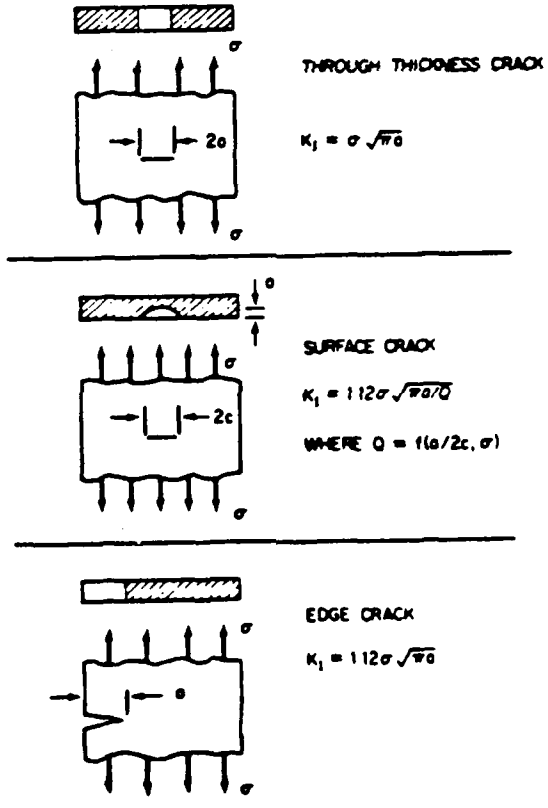


Fig 14: K_I Values for Various Crack Geometries

$$\sigma_{max} = \sigma_m = (1 + \frac{2a}{b})$$

When $b = a$, i.e. a circular hole, $\sigma_m = 3\sigma$. When $b \ll a$, σ_m becomes very large. It is in the limiting case a crack in the plate.

The basic premise of fracture mechanics is the recognition that the actual stress in the structural elements is significantly higher than the nominal stresses calculated by internal loads analysis which did not account for the presence of cracks or flaws. These cracks or flaws were, of course, unintended, but they are introduced by the fabrication of built-up structures. The stress distribution in the vicinity of the crack is generally much higher, and the designer must make sure that they are the sources of failure of the structure. The stresses around and in the vicinity of a crack can best be described by the stress-intensity factors K_I, K_{II} and K_{III}. The subscripts I, II and III refer to the three modes of cracks as shown in Figure 12.

Among these the mode I crack is the one we shall concentrate on. However, the same ideas can be extended to the other two modes of cracks. The mode I crack plays an important role in the design of aircraft elements. The stress-intensity factor K_I can be expressed as a function of the applied nominal stress and the crack length in the case of a through the thickness crack in an infinite plate.

$$K_I = \sigma\sqrt{\pi a} \tag{20}$$

where σ is the nominal stress and a is the semicrack length. If K_I is known, then the stress-distribution in the vicinity of a crack can be expressed by:

$$\sigma_x = \frac{K_I}{(2\pi r)^{1/2}} \cos\frac{\theta}{2}\left[1 - \sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right] \tag{21}$$

$$\sigma_y = \frac{K_I}{(2\pi r)^{1/2}} \cos\frac{\theta}{2}\left[1 + \sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right] \tag{22}$$

$$\tau_{xy} = \frac{K_I}{(2\pi r)^{1/2}} \sin\frac{\theta}{2}\cos\frac{\theta}{2}\cos\frac{3\theta}{2} \tag{23}$$

$$\sigma_z = 0 \qquad (Plane\ Stress) \qquad \tau_{xz} = \tau_{yz} = 0 \tag{24}$$

$$\sigma_z = \nu(\sigma_x + \sigma_y) \qquad Plane\ Strain \tag{25}$$

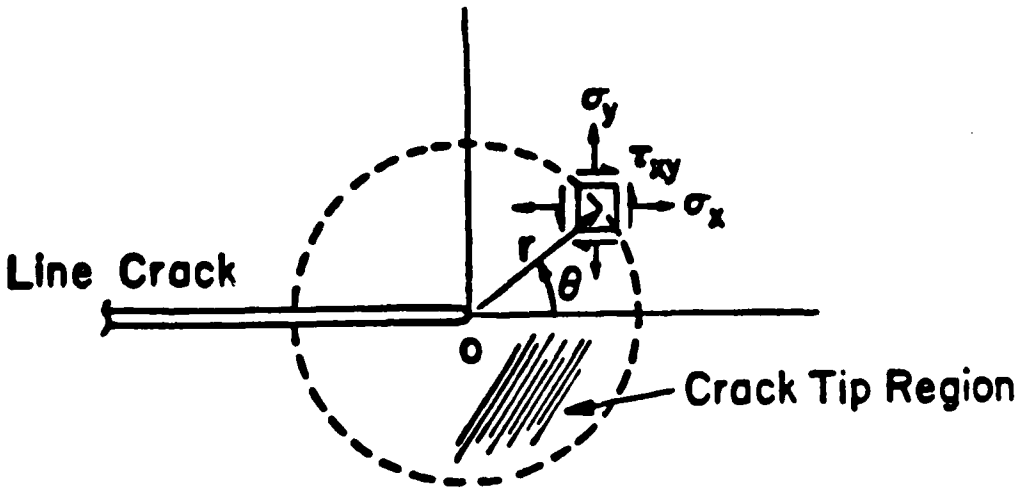


Fig 15: Stress Element Near Crack Tip

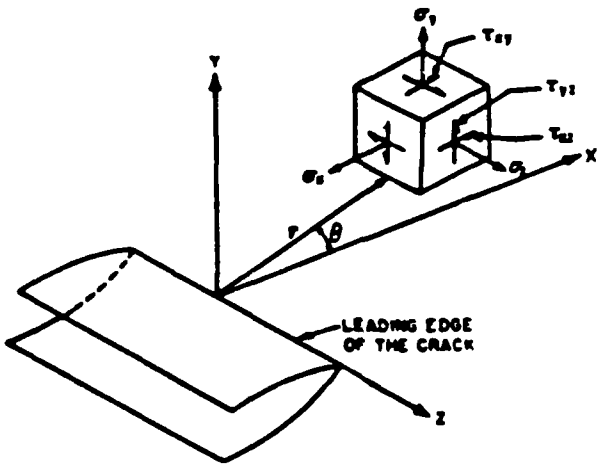


Fig 16: Coordinate System and Stress Components Ahead of a Crack Tip

The exact stress distribution around a crack is not of as much importance to the designer as that of the question of whether this crack precipitates failure (propagates) of the structural element. This concern for failure relates the concepts of critical crack length, critical nominal stress and critical stress intensity factor or fracture toughness of the material.

$$K_{Ic} = \sigma_c \sqrt{\pi a} \tag{26}$$

The critical stress-intensity factor, K_{IC}, which is also referred to as fracture toughness, is a material property and can be determined by standard material tests. Conceptually this procedure is quite simple. Subject a plate with a known crack length and load to failure fracture and determine σ_c for that crack length. Repeat the test with different crack sizes and determine the failure stress.

By repeating this procedure the quantity $\sigma_c \sqrt{\pi a}$, a material constant, can be established and from this, one can determine the fracture toughness (critical stress-intensity factor K_{IC}).

$$K_{IC} = 50\,\text{ksi}\sqrt{\text{in.}} = \sqrt{x}\sigma\sqrt{a}$$

Using this equation, values of the critical crack size for various stress levels are calculated as follows:

$\sigma(ksi)$	$a(in.)$
10	7.96
20	1.99
30	0.88
40	0.50
50	0.32
60	0.22
70	0.16
80	0.12
90	0.10
100	0.08

The allowable stress levels from fracture considerations can be determined by the allowable crack lengths or vice versa when the fracture toughness of the material is known. A more general expression for the stress-intensity factor can be written as

$$K_I = \sigma\sqrt{\pi a}\, f\left(\frac{a}{W}\right) \tag{27}$$

The quantity $f\left(\frac{a}{W}\right)$ accounts for the finite dimensions of the plate. The rate of fatigue crack propagation per cycle can be related to the stress intensity factor as follows:

$$\frac{da}{dN} = f(R, \Delta K) \tag{28}$$

$$R = \frac{K_{min}}{K_{max}} = \frac{S_{min}}{S_{max}} \qquad \Delta K = K_{max} - K_{min} \tag{29}$$

The left hand side of the equation represents the rate of fatigue crack propagation per cycle.

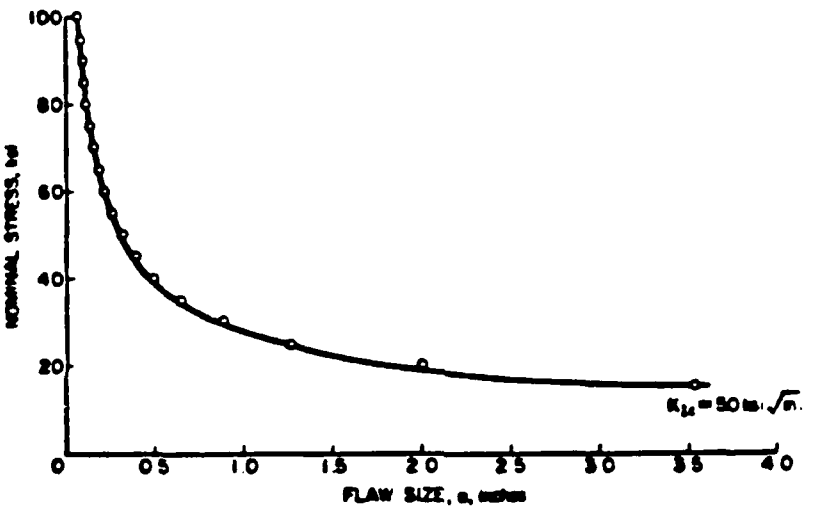


Fig 17: Stress-Flaw-Size Relation for Through-Thickness Crack in Material Having K_{Ic} = 50 ksi$\sqrt{\text{in.}}$

REFERENCES

1. Taylor, J., "Manual on Aircraft Loads", Pergamon Press, Published for AGARD, 1965.

2. Ashley, H., "Engineering Analysis of Flight Vehicles", Addison–Wesley Publishing Co., 1974.

3. Wilkinson, K., et al., "An Automated Procedure for Flutter and Strength Analysis and Optimization of Aerospace Vehicles", Vol. I – Theory, AFFDL-TR-75-137, 1975.

4. Grover, H. J., "Fatigue of Aircraft Structures" Publication of Naval Air Systems Command, NAVAIR 01-1A-13, 1966.

5. Broek, D., "Elementary Engineering Fracture Mechanics", Third Edition, Martinus Nijhoff Publishers, 1982.

OPTIMALITY CRITERIA: A BASIS FOR MULTIDISCIPLINARY DESIGN OPTIMIZATION

Vipperla B. Venkayya
Air Force Wright Aeronautical Laboratories
Wright-Patterson Air Force Base, Ohio 45433-6553

ABSTRACT

This paper presents a generalization of what is frequently referred to in the literature as the optimality criteria approach in structural optimization. This generalization includes a unified presentation of the optimality conditions, the Lagrangian multipliers, and the resizing and scaling algorithms in terms of the sensitivity derivatives of the constraint and objective functions. The by-product of this generalization is the derivation of a set of simple nondimensional parameters which provides significant insight into the behavior of the structure as well as the optimization algorithm. A number of important issues, such as, active and passive variables, constraints and three types of linking are discussed in the context of the present derivation of the optimality criteria approach. The formulation as presented in this paper brings multidisciplinary optimization within the purview of this extremely efficient optimality criteria approach.

INTRODUCTION

In recent years, interest in the multidisciplinary optimization of aerospace structures has been widespread. At present there are many large scale software systems under development both in the U.S. and overseas. Some examples of these are: "ASTROS" [Johnson, Herendeen and Venkayya (1984)] (*A*utomated *S*tructural *O*ptimization *S*ystem being developed for the Air Force Wright Aeronautical Laboratories), "LAGRANGE" [Mikolaj (1987)] (developed by MBB in Germany), "ELFINI" [Petiau and Lecina] (Avions Marcel Dassault in France) and "STAR" [Scion Ltd (1984)] (Royal Aircraft Establishment in UK). A number of other systems are in development around the world. Earlier computer programs like "OPTSTAT" [Venkayya and Tischler (1979)], "ASOP" [Dwyer, Emerton and Ojalvo (1971)], "FASTOP" [Wilkinson, Markowitz, Lerner, George and Batill (1977)], "TSO" [Lynch, Rogers, Braymen and Hertz], "ACCESS" [Schmit and Miura (1976)], etc. have preceded these modern systems, and they have established the feasibility of integrating optimization into structural design. Developers of "MSC NASTRAN" [MacNeal (1971)], "ANSYS" [DeSalvo and Swanson (1985)] and others are actively attempting to incorporate optimization into their systems.

Most of these systems are intended for the preliminary design of aerospace structures using finite element models. The distinguishing feature of these preliminary design systems is that the predicted performance parameters, such as, strength, stiffness, flutter and other aeroelastic parameters, are realizable within a small percentage error. Some of the common disciplines of the integrated design systems are structures, aerodynamics, aeroelasticity, sensitivity analysis and optimization. The next logical step in integration is to include aircraft and spacecraft controls as well.

One of the most challenging problems in structural optimization with finite element models is the ability to handle large order systems with numerous design variables and constraints. The order of the system is defined by the number of degrees of freedom in the analysis. As the order of the system increases, both the response and the sensitivity analysis require excessive computer resources. Since optimization requires several analysis iterations, it is essential that analysis and optimization algorithms be made numerically efficient. Several order reduction and variable linking schemes are available to cope with this computational burden. However, order reduction schemes introduce uncertainty in the accuracy of the analysis. Similarly, variable linking schemes overconstrain the optimization problem. Errors of analysis can propagate, since optimization algorithms are, in general, iterative approaches. Overconstrained optimization problems can only give upper or lower bound solutions depending on the minimization or the maximization problem. Analysis and optimization algorithms that do not depend on order and variable reduction schemes are preferable, if they can efficiently handle the numerical issues.

In a finite element model a structure (continuum) is represented by a large number of discrete (finite) elements. Each element connects a set of grid points. In configuration space each grid point can contribute up to six degrees-of-freedom, three translations and three rotations, to the analysis set. The total number of degrees-of-freedom constitutes the order of the system. The order of the system determines the analysis cost. Similarly, each element of the finite element model contributes one or more (design) variables to the optimization problem. The number of variables increases both the sensitivity analysis and the optimization costs. Since structural design belongs to a class of nonlinear optimization problems, more variables means increased difficulties in obtaining optimal solutions. The limit on most nonlinear programming algorithms in use at the present time is around 100-200 variables. By linking the design variables, one can reduce the problem to a more manageable size and can extend the capabilities of the optimization algorithm to handle large scale systems. Linking is akin to order reduction and, as it was noted earlier, is tantamount to adding more constraints to the system. Moreover, in a large scale system it is not always easy to see the appropriate linking scheme.

In response to the need for the optimization of large practical structures, a discrete

optimality criteria was proposed during the late sixties and early seventies [Venkayya, Khot and Reddy (1969); Venkayya (1971); Venkayya, Khot and Berke (1973)]. This procedure consisted of deriving the optimality conditions and then obtaining the iterative algorithm from the same optimality conditions. This iterative algorithm, together with a scaling procedure, was used to optimize a number of structures with stress, displacement and frequency constraints [Venkayya, Khot and Reddy (1969); Venkayya (1971); Venkayya, Khot and Berke (1973); Venkayya and Tischler ((1983); Grandhi and Venkayya (1987)]. However, the iterative algorithm, the scaling procedure and the Lagrangian multipliers for multiple constraints were derived for each special condition. This approach is not very conducive for optimization in a multidisciplinary setting. Moreover, since most of the applications were in the context of membrane structures, an unintended consensus was that the method is limited to such structures. The purpose of this paper is to generalize this extremely efficient approach and to establish a mathematical basis in the context of a nonlinear programming method. In addition, it is important to dispel the notion that the optimality criteria method has only limited application. The topics to be addressed in this comprehensive derivation are:

a. Optimality conditions

b. Lagrangian multipliers for multiple constraints

c. The iterative algorithm for resizing variables

d. Scaling

e. Active and passive variables

f. Active and passive constraints

g. Linking variables

Then the above conditions will be specialized for the following frequently discussed cases:

a. Displacement constraints - membrane structures

b. Displacement constraints - membrane-bending structures

c. Frequency constraints - membrane-bending structures

d. Stress constraints - membrane-bending structures

e. Scale factor and the nondimensional parameters

The most important topic in this optimality criteria approach is the concept of scaling, and it will be discussed in some detail. The next two important topics are the iterative algorithm together with the specialization of the Lagrangian multipliers. All of these concepts will be derived as a function of the sensitivity derivatives of the constraints and the objective functions. Then this optimization will no longer be addressed in the context

of a single discipline, but instead it will be derived in terms of sensitivity derivatives which can be obtained for all disciplines.

Since sensitivity plays such an important role, it is worthwhile pointing out that there are three different approaches to a sensitivity analysis [Venkayya (1985)]: (a) Taylor's series approximation, (b) adjoint variable or virtual work and (c) finite difference. The first and second approaches are generally efficient, and the finite difference approach is the least efficient. However, the finite difference approach is conceptually the simplest, and it can be used readily in any situation. Throughout this paper it will be assumed that the sensitivity derivatives are available in all disciplines.

OPTIMALITY CONDITIONS

The constrained optimization problem can be stated as follows:

Minimize or maximize the performance function

$$W = W(x_1\ x_2\ \cdots\ x_m) \tag{1}$$

Subject to the constraints

Inequalities

$$Z_j(x_1\ x_2\ \cdots\ x_m) \leq \overline{Z}_j \qquad j = 1, 2, \ldots, k \tag{2}$$

Equalities

$$Z_j(x_1\ x_2\ \ldots\ x_m) = \overline{Z}_j \qquad j = k+1, \ldots, l \tag{3}$$

In addition there are constraints on the variables themselves, and they are defined as

$$\underline{\underline{\underline{x}}} \geq \underset{\sim}{x} \geq \underline{\underline{\underline{x}}} \tag{4}$$

or a subset of $\underset{\sim}{x}$ are assigned fixed values. Functions W (objective or performance) and Z (constraints) are functions of m variables ($x_1 x_2 - - - x_m$), and they will be referred to as design variables or simply variables in the optimization.

The concept of active and passive constraints is defined as follows: a constraint is active if the analysis of the system for a given variable vector shows that $Z_j = \bar{Z}_j$. Otherwise the constraint is considered passive at least in that design. Similarly, a variable is considered active if it is between the bounds defined in Eq 4 and if it was not assigned a fixed value. All other variables are passive.

The constrained optimization problem corresponding to active constraints can be reformulated with a Lagrangian function L as

$$L(\underset{\sim}{x}, \underset{\sim}{\lambda}) = W(\underset{\sim}{x}) - \sum_{j=1}^{p} \lambda_j (Z_j - \overline{Z}_j) \tag{5}$$

where the λ's are the Lagrangian multipliers corresponding to the active constraints. The stationary condition of the Lagrangian function also corresponds to the stationary condition of W

$$\frac{\partial L}{\partial x_i} = \frac{\partial W}{\partial x_i} - \sum_{j=1}^{p} \lambda_j \frac{\partial Z_j}{\partial x_i} = 0 \qquad i = 1, 2, \ldots, m \tag{6}$$

In the above equation all m variables are assumed to be active, and also there are p active constraints. The set of m equations represented by Eq 6 can be written as

$$\sum_{j=1}^{p} e_{ij} \lambda_j = 1 \qquad i = 1, 2, \ldots, m \tag{7}$$

where e_{ij} is the ratio of the sensitivity derivatives of the constraints and the objective function and is given by

$$e_{ij} = \frac{\frac{\partial Z_j}{\partial x_i}}{\frac{\partial W}{\partial x_i}} \tag{8}$$

This quantity, e_{ij}, henceforth will be referred to as the ratio of energy density to weight density or equivalent in the element.

Eqs 7 represent the necessary conditions of optimality, and they are also referred to as Kuhn-Tucker conditions in nonlinear programming. Eqs 7 in matrix form can be written as

$$\underset{\sim}{e}\underset{\sim}{\lambda} = \underset{\sim}{1} \tag{9}$$

where $\underset{\sim}{e}$ is an $m \times p$, $\underset{\sim}{\lambda}$ a $p \times 1$ and $\underset{\sim}{1}$ a $m \times 1$ matrix. Premultiplying both sides of Eq 9 by $\underset{\sim}{e}^t \bar{\underset{\sim}{A}}$ gives

$$\underset{\sim}{e}^t \bar{\underset{\sim}{A}} \underset{\sim}{e} \underset{\sim}{\lambda} = \underset{\sim}{e}^t \bar{\underset{\sim}{A}} \underset{\sim}{1} = \bar{\bar{\underset{\sim}{Z}}} \tag{10}$$

where the weighting matrix $\bar{\underset{\sim}{A}}$ is an $m \times m$ diagonal matrix. The elements of $\bar{\underset{\sim}{A}}$ will be selected such that the elements of $\bar{\bar{\underset{\sim}{Z}}}$ will represent some energy or equivalent in the system.

One of the important requirements of $\bar{\underset{\sim}{A}}$ is that it be positive definite. It should also be noted that an interesting generalization of the optimality criterion can be derived from the selection of an appropriate $\bar{\underset{\sim}{A}}$. The implication being that through the weighting matrix $\bar{\underset{\sim}{A}}$ the method can be extended beyond structural optimization. In structural optimization problems the elements of the diagonal matrix $\bar{\underset{\sim}{A}}$ are assumed to be the weights of the individual structural elements. Then the elements $\bar{\bar{Z}}_j$ are given by

$$\bar{\bar{Z}}_j = \sum_{i=1}^{m} e_{ij} \bar{A}_{ii} \qquad j = 1, 2, \ldots, p \tag{11}$$

As stated previously the number p corresponds to the active set of constraints. Now Eq 10 can be written as

$$\underset{\sim}{H}\underset{\sim}{\lambda} = \bar{\bar{\underset{\sim}{Z}}} \tag{12}$$

Eqs 12 are a nonlinear set of equations. Since the elements of $\underset{\sim}{H}$ are functions of the primary variables $\underset{\sim}{x}$, which are themselves unknown, the solution of Eqs 12 for unknown λ's can be determined by Newton-Raphson or other approximate methods. These iterative methods converge only if the starting solution is close to the actual solution. Also in the absence of a unique solution for the λ's it would be difficult to select a reasonable initial solution. To avoid these difficulties a simpler, but an approximate method, was proposed in 1973 [Venkayya, Khot and Berke (1973)].

LAGRANGIAN MULTIPLIERS FOR MULTIPLE CONSTRAINTS

The method for estimating the Lagrangian multipliers is based on a very simple concept. They are determined by invoking the condition of a single active constraint. Then the resulting λ's are used as weighting parameters in a multiconstraint problem. Since these parameters will be updated in each cycle of the iteration, this method works as well as any other approximate method. Basically, this assumption implies that the $\underset{\sim}{H}$ in Eq 12 is strongly diagonal. This may not be true, but should not deter the use of a single constraint approximation. Approximations cannot be avoided in any method of determining the Lagrangian multipliers because of the nonlinearities. Another advantage of this approach is that by monitoring the Lagrangian multipliers, one can well assess the behavior of the constraints and predict how the design progresses to the optimum. This ability to predict behavior is essential in order to eliminate significant anomalies and uncertainties.

For a single constraint case the m equations of optimality can be written as

$$e_1\lambda = 1 \qquad e_2\lambda = 1 \quad \cdots \quad e_m\lambda = 1 \tag{13}$$

It is evident from Eqs 13 that this condition at the optimum can only be true when

$$e_1 = e_2 = \cdots = e_m = e \tag{14}$$

and

$$\lambda = \frac{1}{e} \tag{15}$$

Now Eq 10 can be written as

$$e(\underset{\sim}{1}^t \overline{\underset{\sim}{A}}\,\underset{\sim}{1}) = \overline{\overline{Z}} \tag{16}$$

If a quantity $\bar{W}$ is defined as

$$\overline{W} = \underset{\sim}{1}^t \overline{\underset{\sim}{A}}\,\underset{\sim}{1} \tag{17}$$

then from Eq 16 e becomes

$$e = \frac{1}{\lambda} = \frac{\overline{\overline{Z}}}{\overline{W}} \tag{18}$$

or

$$\lambda = \frac{\overline{W}}{\overline{\overline{Z}}} \tag{19}$$

For multiple constraints the approximation is

$$\lambda_j = \frac{\bar{W}}{\bar{\bar{Z}}_j} \tag{20}$$

The meaning of the parameter $\bar{W}$ depends on what is selected for the weighting matrix $\underset{\sim}{\bar{A}}$. For example, in structural weight minimization problems the weight of each element in the finite element model can be selected as the diagonal elements of $\underset{\sim}{\bar{A}}$. In that case $\bar{W}$ is simply the total weight of the structure, and $\bar{\bar{Z}}$ is the imposed constraint or a function of it. However, one should be cautioned that Eq 20 is not limited to weight minimization problems, because nowhere in its derivation was this requirement invoked.

ITERATIVE ALGORITHM (RESIZING ALGORITHM)

The optimality condition as defined by Eq 7 states that at the optimum the weighted sum of the energy density (or equivalent) to the weight density ratio corresponding to the active constraints must be the same in all the finite elements in the structure. The weighting parameters are the Lagrangian multipliers. Now the iterative algorithm can be derived by multiplying both sides of Eq 7 by x_i^α

$$x_i^\alpha = x_i^\alpha \left[\sum_{j=1}^{p} e_{ij} \lambda_j \right] \tag{21}$$

Eq 21 can also be written as

$$x_i = x_i \left[\sum_{j=1}^{p} e_{ij} \lambda_j \right]^{\frac{1}{\alpha}} \tag{22}$$

Then the resizing formula can be written as

$$x_i^{\nu+1} = x_i^\nu \left[\sum_{j=1}^{p} e_{ij} \lambda_j \right]^{\frac{1}{\alpha}} \tag{23}$$

where α is defined as a step size parameter. A large value of α represents a smaller step size and vice-versa. For most problems $\alpha = 2$ represents a reasonable step size, because it assures a reasonable rate of convergence. However, as the design approaches the optimum, there is an increasing possibility of constraint switching and other anomalies which can disturb a smooth convergence. When such conditions are encountered, the value of α can be increased to reduce the step size and capture the optimum design. In fact, by monitoring the single constraint approximation of the Lagrangian multipliers, one can easily predict when the value of α needs to be increased from 2. For most problems an α value of 2 is ideal for the first 80 to 90% of the iterations. Any increase in the α value is necessary (not always) only in the last 10 to 20% of the iterations. In these instances a change over to an α value of 3 or 4 is adequate. In summary, it should be pointed out that a larger value of

α increases the number of iterations but provides a smoother convergence. By the same token small values of $\alpha(\not< 1)$ speed up the iteration but can miss the optimum because of constraint switching or other anomalies.

The iterative algorithm, as defined by Eq 23, is distinctly different from the standard nonlinear programming algorithms which define

$$x_i^{\nu+1} = x_i^{\nu} + \alpha_i^{\nu} D_i^{\nu} \tag{24}$$

where α represents the step size and D represents the direction of travel. Both α and D are generally constructed from the sensitivity derivatives, e, as in the optimality criteria approach.

The difference in philosophy of the two resizing approaches represented by Eqs 23 and 24 is quite significant and can be explained with the help of the two variable design space in Fig. 1. In the nonlinear programming approach, Eq 24, the search is from point to point in the design space. The computational effort and the number of cycles of iteration become very large when the number of variables increases. This observation is a result of over 30 years of experience reported in the literature. If the number of variables exceeds 100-200, these algorithms can hardly give reasonable solutions. The search, as represented by Eq 23 on the other hand, sweeps through the design space as indicated in Fig. 1 and tends to be insensitive to the number of design variables. The resizing procedure, as defined in Eq 23, together with the scaling procedure to be outlined in the next section are described as the optimality criteria approach in structural design.

SCALING PROCEDURE

The scaling procedure can be explained with the help of two designs as represented by the two variable vectors $\underset{\sim}{x}$ and $\bar{\underset{\sim}{x}}$. Now the relationship between the two variable vectors is given by

$$\bar{\underset{\sim}{x}} = \Lambda \underset{\sim}{x} \tag{25}$$

where Λ is a single scalar parameter which will be referred to as a scale factor. ($\Lambda > 0$). If $\underset{\sim}{dx}$ is the difference vector between the two designs, then one can write

$$\underset{\sim}{dx} = \bar{\underset{\sim}{x}} - \underset{\sim}{x} = (\Lambda - 1)\underset{\sim}{x} \tag{26}$$

Also if R and $\bar{R}$ are the response quantities respectively in the two designs, then a change in response can be represented by

$$dR = \bar{R} - R \tag{27}$$

Now from the definition of the total differential (first order approximation of the Taylor's Series) the following relationship can be written

$$dR = \frac{\partial R}{\partial x_1} dx_1 + \frac{\partial R}{\partial x_2} dx_2 + \cdots + \frac{\partial R}{\partial x_m} dx_m \tag{28}$$

Then dR can also be written as (from Eqs 26 and 28)

$$dR = (\Lambda - 1) \sum_{i=1}^{m} \frac{\partial R}{\partial x_i} x_i \tag{29}$$

Then

$$\frac{dR}{R} = (\Lambda - 1) \frac{\sum_{i=1}^{m} \frac{\partial R}{\partial x_i} x_i}{R} \tag{30}$$

An examination of Eq 30 presents two interesting cases.

CASE 1:

$$\frac{\sum_{i=1}^{m} \frac{\partial R}{\partial x_i} x_i}{R} \leq 0 \tag{31}$$

In this case a new parameter μ is defined as

$$\mu = - \frac{\sum_{i=1}^{m} \frac{\partial R}{\partial x_i} x_i}{R} \tag{32}$$

Then Eq 30 can be written as

$$\frac{dR}{R} = (1 - \Lambda)\mu \tag{33}$$

Now the scale factor Λ can be written as

$$\Lambda = 1 - \frac{dR}{R} \frac{1}{\mu} = 1 - b \tag{34}$$

where

$$b = \frac{1}{\mu} \frac{dR}{R} \qquad b \ll 1 \tag{35}$$

Eq 34 can also be written as

$$\frac{1}{\Lambda} = \frac{1}{1 - b} = 1 + b \tag{36}$$

by neglecting the higher order terms of b in a binomial expansion. Now dR/R can be written as

$$\frac{dR}{R} = \frac{\mu}{\Lambda} - \mu \tag{37}$$

Adding 1 to both sides of Eqs 37 one can write

$$\frac{R + dR}{R} = \frac{\mu}{\Lambda} - \mu + 1 \tag{38}$$

A new parameter, β, which will be referred to as the target response ratio, is defined as

$$\beta = \frac{\text{New Response } (\bar{R})}{\text{Initial Response } (R)} = \text{Target Response Ratio} \tag{39}$$

Then

$$\beta = \frac{\mu}{\Lambda} - \mu + 1 \tag{40}$$

Solving for the scale factor Λ

$$\Lambda = \frac{\mu}{\beta + \mu - 1} \tag{41}$$

CASE 2:

$$\frac{\sum_{i=1}^{m} \frac{\partial R}{\partial x_i} x_i}{R} > 0 \tag{42}$$

Now the parameter μ is defined as

$$\mu = \frac{\sum_{i=1}^{m} \frac{\partial R}{\partial x_i} x_i}{R} \tag{43}$$

Then the scale factor Λ can be written as

$$\Lambda = \frac{\beta + \mu - 1}{\mu} \tag{44}$$

An examination of Eqs 41 and 44 reveals some interesting facts:

1. In CASE 1 the scale factor is inversely proportional to the target response ratio, and in CASE 2 it is directly proportional to β.
2. The response of the system, R, and the response sensitivity, $\partial R/\partial x_i$, can be determined from an analysis of the system for a given variable vector $\underset{\sim}{x}$. The target response (or desired response) can be determined from the constraint definition. Then the target response ratio, β, and the parameter, μ, are known. Then the scale factor Λ can be determined explicitly for any type of structure and constraints.
3. Both β and μ are non-dimensional parameters, and their range can be estimated quite well for a given structure and constraints. For example, if the desired (target) response is 20% greater than the original response, then β would be 1.2. For displacement constraints in membrane structures $\mu = 1$, and Eq 41 becomes

$$\Lambda = \frac{1}{\beta} \tag{45}$$

This means that the scale factor is inversely proportional to the target response ratio. The relationship described in Eq 45 is exact. The following sections will discuss additional details.

ACTIVE AND PASSIVE VARIABLES

The definition of active and passive variables was given in Section 2 as part of the formulation of the optimization problem. All those variables that are free to participate in the optimization are called active variables. The variables on that part of the structure that are not allowed to change and those beyond the range defined by the side constraints, Eq 4, are the passive variables. There is always the question of why these passive variables should be treated as variables at all, if they do not participate in the optimization. Even though these variables are not changing in absolute terms, they are changing relative to the active variables. This relative change does effect the response and the sensitivity of the structure.

The effect of the distinction between the active and passive variables on the optimization problem formulation and solution is explained by citing specific equations. (a) For example, the optimality condition as defined by Eqs 7 or 9 is not affected by this distinction. In other words even though the active variables are only a subset of the m variables, they all participate in the optimality condition. The energy density or equivalent as defined by Eq 8 remains the same. (b) The Lagrangian multipliers as defined by Eqs 12 or 20 are also uneffected. (c) The resizing algorithm, as defined by Eq 23, applies only to the active variables which means the passive variables are not resized. (d) In determining the scale factor Λ by Eqs 41 or 44, only the active variables are included in the summation. The parameter μ, as defined by Eqs 32 or 43, includes only the active variables in the summation also.

ACTIVE AND PASSIVE CONSTRAINTS

The concept of active and passive constraints was the most obvious and simplest concept when it was proposed [Venkayya, Khot and Reddy (1969); Venkayya (1971); Venkayya, Khot and Berke (1973)]. This concept led to the constraint deletion techniques in the structural applications of nonlinear programming algorithms. The way this concept is used in the optimality criteria is explained here for further clarification.

The target response ratio as defined in Eq 39 is invoked here for this explanation. The target response ratio is the ratio of the imposed constraint value to the value of the constraint determined in the analysis. In each iteration (analysis) the target response ratios can be determined (a trivial task) for all the constraints. An array of β^s is generated in this process ($\beta > 0$). Now the active constraints can be defined as

$$Active\ \ Constraints = p = p_E + p_I$$

where p_E represents all the equality constraints (Eq 3) and p_I represents the constraint set derived from the inequalities (Eq 2). All the constraints with the lowest value of β (the greatest value in the case of inequalities expreseed as $\geq$) and its vicinity contribute to the set p_I. This constraint set can change (need not be the same) in each iteration.

The criticism that the active constraint set at the optimum must be known in advance in order to apply the optimality criteria approach is not true. The active constraint set is defined just for that iteration, and the algorithm itself eventually drives the design to the active constraint set at the optimum.

LINKING VARIABLES

As discussed in the introduction, linking of variables is often used to reduce the order of the design space. This is acceptable as long as it is recognized that linking is tantamount to adding additional constraints which can affect the optimum solution. However, linking of variables can be very effective in practical designs, if it is done after a thorough examination of unlinked designs. By comparing the linked and unlinked designs, one can assess the price of linking. Sometimes the performance demands of modern aerospace systems and the recent developments in computer controlled manufacturing processes may accommodate the unlinked designs or reduce the linking to a minimum.

There are three types of linking and all of them have a similar effect on the optimization algorithm.

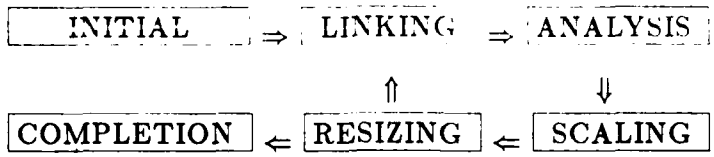
a. The simplest case of linking is to assign a single variable to a group of elements. This means that all the elements in that group will have the same variable value.

b. Linking by polynomial variation is another option. This involves the selection of a group of elements based on (possibly) their spatial location and linking them by linear, quadratic or cubic polynomials. The variables in the polynomials are parameters that determine the location. This concept was used very effectively in programs like TSO [Lynch Rogers, Braymen and Hertz]. Since the structure is represented by a single trapezoidal flat surface in the TSO program, the meaning of polynomial linking is quite simple and appealing. However, it can easily be generalized to three dimensional finite element models as shown later in this section.

c. Shape function linking is essentially an extension of polynomial linking, but its application becomes meaningful only to a more sophisticated user.

A more detailed discussion of linking in the context of the present optimality criteria approach is presented here. Linking does not affect the optimality conditions or the expressions for the Lagrangian multipliers. It does not even affect the scaling. Here linking is not used to reduce the size of the design space, as the dimensionality is not of much consequence in the optimality criteria approach. It is essentially intended for the purpose of tailoring optimum designs to manufacturing requirements and not for accommodating algorithm limitations.

The linking algorithm is introduced upfront as an independent operation in the optimization as shown in the schematic diagram.



Design Scheme With Linking

The basic linking algorithm is explained in the context of the general transformation

$$\underset{\sim}{x} = \underset{\sim}{T}\bar{\underset{\sim}{x}} \tag{46}$$

where $\underset{\sim}{x}$ is the $m \times 1$ variable vector that goes into the analysis. The vector $\bar{\underset{\sim}{x}}$ is an $\ell \times 1 (\ell \leq m)$ reduced variable vector. This vector is a subset of the initial design the first time, and then a subset of the vector coming from the resizing algorithm. The transformation matrix $\underset{\sim}{T}$ is an $m \times \ell$ matrix. The three linking schemes discussed earlier can be accommodated in the definition of the transformation matrix.

a. Assigning single variables to groups of elements:

   The variable vector $\underset{\sim}{x}$ is represented by ℓ groups and each group contains one or more variables. All the variables in each group have the same value. This value will be the largest variable in that group coming from resizing. Thus the transformation matrix in this case is given as

$$\underset{\sim}{T}^t = \begin{pmatrix} \underset{\sim}{T}_1^t & 0 & 0 \\ 0 & \underset{\sim}{T}_2^t & 0 \\ 0 & 0 & \underset{\sim}{T}_3^t \end{pmatrix} \tag{47}$$

   where $\underset{\sim}{T}_1$, $\underset{\sim}{T}_2$ and $\underset{\sim}{T}_3$ are submatrices with dimensions corresponding to the number of variables in each group. If the number of variables in the groups are the same, then

$$\underset{\sim}{T}_1^t = \underset{\sim}{T}_2^t = \underset{\sim}{T}_3^t = [1\ 1\ 1\ \cdots\ 1] \tag{47}$$

b. Polynomial variation of the elements in each group:

   The transformation matrix can be modified by simply replacing the ones by coefficients of the poynomial. If it is a linear linking, it involves two variables, three in the case of quadratic linking and so on. A shifting procedure as explained in the shape function linking can select an effective subset from the resized variables.

c. Shape function linking involves a fully populated transformation matrix.

The following steps outline the iterative scheme for shape function linking.

1. Select the number of groups, ℓ.

2. Select an appropriate number of elements from the initial or resized vector in descending order ($\underset{\sim}{\bar{x}}^o$ is a subset of $\underset{\sim}{\bar{x}}$).

3. Substitute the variables selected in step 2 into the transformation equation and determine the intermediate vector $\underset{\sim}{\bar{\bar{x}}}$.

4. Shift the $\underset{\sim}{\bar{\bar{x}}}$ vector such that

$$\bar{x}_i^{\nu+1} = \bar{\bar{x}}_i^{\nu} \pm \Delta x \tag{48}$$

where Δx is defined as follows:

CASE 1 : Any $(x_i - \bar{x}_i) < 0 \qquad i = 1, 2, \ldots, m$

$$then \qquad \Delta x = max\ |\bar{\bar{x}}_i - \bar{x}_i| \qquad from\ the\ set\ (\bar{\bar{x}}_i - \bar{x}_i) < 0 \tag{49}$$

CASE 2 : All $(\bar{\bar{x}}_i - \bar{x}_i) \geq 0 \qquad i = 1, 2, \ldots, m$

$$then \qquad \Delta x = min\ (\bar{\bar{x}}_i - \bar{x}_i) \tag{50}$$

5. Now replace $x^{\nu} = x^{\nu+1}$

6. Repeat steps 2 to 5 until

$$\bar{x}^{\nu+1} = \bar{x}^{\nu} = x \tag{51}$$

The advantage of this linking procedure is that it leaves the remaining optimization algorithm untouched.

SPECIALIZATION TO SPECIFIC DESIGN CONDITIONS

A number of issues related to optimization by an optimality criteria approach were addressed in general terms using sensitivity derivatives. The purpose of this section is to examine, in more detail, the implications when the method is applied to specific design conditions. The following design conditions are examined in the context of structural weight minimization.

a. Displacement constraints - membrane structures

b. Displacement constraints - membrane-bending structures

c. Frequency constraints - membrane-bending structures

d. Stress constraints - membrane-bending structures

e. Scale factor and the nondimensional parameters

The optimality conditions (Eqs 7 or 9), the expressions for the Lagrangian multipliers (Eqs 12 and 20), and the resizing algorithm (Eq 23) are discussed briefly when applied to these special design conditions. However, a more detailed examination of the scaling procedure, in light of these special conditions, provides fascinating information on the overall behavior of the structure in optimization.

a. Displacement Constraints - Membrane Structures

This specialization is addressed in the context of structural weight minimization. A brief examination of the optimality conditions (Eqs 7 or 9), the Lagrangian multipliers (Eq 20), the resizing algorithm (Eq 23), and the scale factor (Eqs 41 or 44) would provide more tangible details.

In a finite element model the structural weight is defined as (the objective function W in Eq 1)

$$W = \sum_{i=1}^{m} \rho_i x_i l_i \tag{52}$$

where W is a linear function of the variables x_i. The product $x_i \ell_i$ is the volume of the element, and ρ_i is the weight density of the material. The applied load vector $\underset{\sim}{P}$ and the resulting displacement vector $\underset{\sim}{u}$ are related by

$$\underset{\sim}{P} = \underset{\sim}{K}\,\underset{\sim}{u} \tag{53}$$

The displacement constraint Z_j in Eq 2 can be written as

$$Z_j = u_j = \underset{\sim}{F}_j^t \underset{\sim}{u} \tag{54}$$

where $\underset{\sim}{F}_j$ is the virtual load vector in which $F_j = 1$ for $i = j$ and $F_j = 0$ when $i \neq j$. The displacement u_j is the active constraint. The quantity e_{ij} in the optimality condition, Eqs 7 or 9 becomes [Venkayya, Khot, Berke (1973)].

$$e_{ij} = -\frac{\underset{\sim}{f}_j^t \underset{\sim}{K}_i \underset{\sim}{u}}{\rho_i x_i l_i} \tag{55}$$

where $\underset{\sim}{f}_j$ is the virtual displacement vector corresponding to the load vector $\underset{\sim}{F}_j$, and $\underset{\sim}{K}_i$ is the i^{th} element stiffness matrix in the global coordinate system.

If the diagonal elements of the matrix $\underset{\sim}{A}$ in Eq 10 are selected as the weight of the structural elements in the finite element model, then one can write the relation

$$\bar{\bar{Z}} = \bar{Z} \tag{56}$$

and

$$\bar{W} = W \tag{57}$$

where $\bar{Z}$ is the constrained value of the displacement. Then the Lagrangian multiplier is simply the ratio of the current weight of the structure and the constrained value of the active displacement.

$$\lambda_j = -\frac{W}{\bar{Z}_j} \tag{58}$$

W and $\bar{Z}_j$ are known and there is no need for special computations for λ_j. With the above definitions the resizing algorithm, Eq 23, does not need further clarification.

The scale factor as defined in Section 2 requires the parameter μ which is defined as

$$\mu = \pm \frac{\sum_{i=1}^{m} \frac{\partial R}{\partial x_i} x_i}{R} \tag{59}$$

The response quantity, R, in this case is the displacement at a point that is active with respect to the constraint definition.

$$R = u_j = \underset{\sim}{F}_j^t \underset{\sim}{u} \tag{60}$$

Substitution of Eqs 53 and 60 in Eq 59 gives the expression for μ as

$$\mu = \frac{\underset{\sim}{f}_j^t \underset{\sim}{K} \underset{\sim}{u}}{\underset{\sim}{F}_j^t \underset{\sim}{u}} = 1 \tag{61}$$

where the virtual displacement vector $\underset{\sim}{f}$ is given by the relation

$$\underset{\sim}{F}_j = \underset{\sim}{K} \underset{\sim}{f}_j \tag{62}$$

Then the scale factor is simply (Eq. 41)

$$\Lambda = \frac{1}{\beta} \tag{63}$$

Eq 63 is the classic result (without approximations) for membrane structures with displacement constraints. This equation simply says that the scale factor is inversely proportional to the target response ratio.

b. Displacement Constraints - Membrane-Bending Structures

In a plane frame structure each element of the structure has two variables. These are the cross-sectional area, x_i, and the moment of inertia, I_i. They are never really completely independent variables, because it may not be possible to build an element in such a case. The most general relationship that can be assumed is

$$I_i = d_i x_i^{n_i} \tag{64}$$

where d_i and n_i are constants. Both d_i and n_i can be different for different elements. The value of n_i for most hollow box beams and I-beams can be approximated as

$$1 \leq n_i \ll 2 \tag{65}$$

For solid rectangular beams this value would be approximately

$$n_i \simeq 3 \tag{66}$$

For all other sections $n_i < 3$.

The quantity e_{ij} in the optimality condition takes the form

$$e_{ij} = -\frac{\underset{\sim}{f}_j(\underset{\sim}{K}_{Ai} + n_i \underset{\sim}{K}_{Bi})\underset{\sim}{u}}{\rho_i x_i l_i} \tag{67}$$

where K_{Ai} and K_{Bi} are the element axial and bending stiffnesses in the global coordinate system. The Lagrangian multipliers are given by

$$\lambda_j = -\frac{W}{\overline{Z}(\mu_{Aj} + \mu_{Bj})} \tag{68}$$

where the parameters μ are defined as

$$\mu_{Aj} = \frac{\sum_{i=1}^{m} \underset{\sim}{f}_j^t \underset{\sim}{K}_{Ai} \underset{\sim}{u}}{\underset{\sim}{F}_j^t \underset{\sim}{u}} \tag{69}$$

$$\mu_{Bj} = \frac{\sum_{i=1}^{m} n_i \underset{\sim}{f}_j^t \underset{\sim}{K}_{Bi} \underset{\sim}{u}}{\underset{\sim}{F}_j^t \underset{\sim}{u}} \tag{70}$$

The parameter μ in the scale factor definition (Eqs 32 or 43) can be written as

$$\mu = \mu_{Aj} + \mu_{Bj} \tag{71}$$

The vectors $\underset{\sim}{F}_j$ and $\underset{\sim}{f}_j$ are the virtual load and displacement vectors, respectively, as defined earlier (Eqs 54 and 62). Then the scale factor becomes

$$\Lambda = \frac{\mu_{Aj} + \mu_{Bj}}{\beta + \mu_{Aj} + \mu_{Bj} - 1} \tag{72}$$

An examination of Eq 72 in the light of three special cases provides an interesting insight.

a. For truss or membrane structures

$$\mu_{Aj} = 1 \qquad \mu_{Bj} = 0 \tag{73}$$

Then the scale factor is inversely proportional to the target response ratio as noted earlier.

b. For membrane bending structures with $n_i = n = 1$

$$\mu_{Aj} + \mu_{Bj} = 1 \tag{74}$$

Then again the scale factor (Λ) is inversely proportional to the target response ratio (β).

c. For membrane-bending structures with $n_i > 1$, the value of μ can be described as

$$\mu = \mu_{Aj} + \mu_{Bj} \geq 1 \quad for \quad n_i \geq 1 \tag{75}$$

However, the limits on μ are $1 \leq \mu \leq 3$. Additional comments on the behavior of the parameters μ_{Aj} and μ_{Bj} and the optimization algorithm are given in the last section. It should be noted that $n_i < 1$ has little meaning in practical structures.

c. Frequency Constraints - Membrane-Bending Structures

The constraint in this case is ω^2 (ω is the circular frequency) which means

$$Z = \omega^2 \tag{76}$$

The quantity e_{ij} in the optimality condition becomes

$$e_{ij} = \frac{\underset{\sim}{\phi}_j^t(\underset{\sim}{K}_{Ai} + n_i\underset{\sim}{K}_{Bi})\underset{\sim}{\phi}_j - \omega_j^2\underset{\sim}{\phi}_j^t\underset{\sim}{M}_{si}\underset{\sim}{\phi}_j}{\underset{\sim}{\phi}_j^t\underset{\sim}{M}\underset{\sim}{\phi}_j\rho_i x_i l_i} \tag{77}$$

where $\underset{\sim}{K}_{Ai}$ and $\underset{\sim}{K}_{Bi}$ are the axial and bending stiffnesses of the i^{th} element. $\underset{\sim}{M}_{Si}$ is the structural mass of the i^{th} element. The Lagrangian multiplier becomes

$$\lambda_j = \frac{W}{\overline{Z}_j(\mu_{Aj} + \mu_{Bj} - \eta_j)} \tag{78}$$

The scale factor in terms of the target response ratio can be written as

$$\Lambda = \frac{\mu_{Aj} + \mu_{Bj} - 1 + \gamma_j\beta_j^2}{\mu_{Aj} + \mu_{Bj} - \beta_j^2\eta_j} \tag{79}$$

where μ_{Aj} and μ_{Bj} are the axial and bending modal stiffness ratios, and they are defined as

$$\mu_{Aj} = \frac{\sum_{i=1}^{m}\underset{\sim}{\phi}_j^t\underset{\sim}{K}_{Ai}\underset{\sim}{\phi}_j}{\underset{\sim}{\phi}_j^t\underset{\sim}{K}\underset{\sim}{\phi}_j} \tag{80}$$

$$\mu_{Bj} = \frac{\sum_{i=1}^{m} n_i\underset{\sim}{\phi}_j^t\underset{\sim}{K}_{Bi}\underset{\sim}{\phi}_j}{\underset{\sim}{\phi}_j^t\underset{\sim}{K}\underset{\sim}{\phi}_j} \tag{81}$$

The parameters γ_j and η_j are the modal nonstructural and structural mass ratios respectively

$$\gamma_j = \frac{\underset{\sim}{\phi}_j^t\underset{\sim}{M}_c\underset{\sim}{\phi}_j}{\underset{\sim}{\phi}_j^t\underset{\sim}{M}\underset{\sim}{\phi}_j} \tag{82}$$

$$\eta_j = \frac{\underset{\sim}{\phi}_j^t\underset{\sim}{M}_s\underset{\sim}{\phi}_j}{\underset{\sim}{\phi}_j^t\underset{\sim}{M}\underset{\sim}{\phi}_j} \tag{83}$$

where M_s and M_c are the structural and nonstructural mass matrices. The relationship between η_j and γ_j is

$$\eta_j + \gamma_j = 1 \tag{84}$$

The target response ratio β_j^2 is defined as

$$\beta_j^2 = \frac{\omega_{jn}^2}{\omega_{jo}^2} \tag{85}$$

where ω_{jn} and ω_{jo} are the new and the initial circular frequencies respectively. The subscript j refers to the mode shape number.

An examination of Eqs 77 to 81 reveals a number of interesting facts:

1. For structures with only membrane elements

$$\mu_{Aj} = 1 \qquad \mu_{Bj} = 0 \tag{86}$$

Then the scale factor can be written as

$$\Lambda = \frac{\gamma_j \beta_j^2}{1 - \eta_j \beta_j^2} \tag{87}$$

This is the same result that was derived in 1983 [Venkayya and Tischler (1983)].

2. For structures with membrane bending elements such that

$$n_i = n = 1 \tag{88}$$

the parameters μ_{Aj} and μ_{Bj} satisfy the relation

$$\mu_{Aj} + \mu_{Bj} = 1 \tag{89}$$

Then the scale factor relation is once again the same as that given in Eq 87.

3. For structures with membrane-bending elements that satisfy Eq 64 but the j^{th} mode shape predominantly excites only the axial stiffness, then

$$\mu_{Aj} \simeq 1 \qquad \mu_{Bj} = 0 \tag{90}$$

The behavior reverts to case 1.

4. If the mode shape predominantly excites the bending stiffness only and also Eq 88 is satisfied, then

$$\mu_{Aj} \simeq 0 \qquad \mu_{Bj} = 1 \tag{91}$$

Again the scale factor equation is the same as Eq 87.

5. For structures with membrane-bending elements but n_i is bound by

$$1 \leq n_i \leq 3 \tag{92}$$

then the μ parameter limits can be written as

$$\mu_{Aj} + \mu_{Bj} \leq 3 \tag{93}$$

n_i values beyond the limits defined in Eq 92 have no meaning in terms of a physical structure, and the μ parameter has a maximum limit of 3. Then the limiting relationships for the scale factor are Eq 87 and

$$\Lambda = \frac{2 + \gamma_j \beta_j^2}{3 - \beta_j^2 \eta_j} \tag{94}$$

6. The effect of the parameter $\beta_j^2 \eta_j$ are such that its limits are

$$0 \leq \beta_j^2 \eta_j < 1 \tag{95}$$

for Eq 87 and

$$0 \leq \beta_j^2 \eta_j < 3 \tag{96}$$

for Eq 94.

Values of $\beta_j^2 \eta_j$ beyond the limits specified by Eqs 95 and 96 have no meaning.

For low values of $\beta_j^2 \eta_j$ the scale factor predictions will be very good. As the parameter reaches the upper bound, the scale factor predictions deteriorate, not because of the approximations involved, but due to the inherent illconditioning in the problem (See Eqs 87 and 94). It is safe to say that if $\beta_j^2 \eta_j > 2/3$ in Eq 95 and > 2 in Eq 96, then the scaling has to be done in two steps (by reducing the value of β) which means an additional analysis in the cycle. The physical meaning of these statements can be explained by examining the two extreme cases:

a. The structural mass is very small compared to the nonstructural mass

$$\eta_j \ll 1 \quad or \quad \eta_j \simeq 0 \qquad (\gamma_j = 1) \tag{97}$$

Then $\beta_j^2 \eta_j \simeq 0$ and the scale factor is directly proportional to the target response rato β_j^2. Predictions are extremely good.

b. The structural mass is dominant and there is no significant nonstructural mass

$$\eta_j \simeq 1 \qquad \gamma_j = 0 \tag{98}$$

In such a case, for the scale factor solution to be non-trivial, the denominator must be equal to zero.

$$\beta_j^2 \eta_j = 1 \tag{99}$$

If Eq 99 is true, then $\beta_j^2 = 1$, because η_j is already assumed to be one, which means no scaling is possible when the nonstructural mass is zero. In real aerospace structures the structural mass contribution seldom exceeds 20 to 30%. So it is not difficult to limit the values of $\beta_j^2 \eta_j < 2/3$ in Eq 87 and 2 in Eq 94 and avoid a second analysis for scaling.

In summary, it must be stated that by monitoring the parameters μ_{Aj}, μ_{Bj}, and η_j or (γ_j), one can predict the behavior of the iterative optimization algorithm extremely well and avoid any aberrations.

d. Stress-Constraints - Membrane-Bending Structures

Once again the relationship between x and I is assumed to be

$$I_i = d_i x_i^{n_i} \tag{100}$$

Now the stress in a given member is written as

$$\sigma_j = \underset{\sim}{T}_j^t \underset{\sim}{S}_j \tag{101}$$

where the vector $\underset{\sim}{T}_j$ is defined as

$$\underset{\sim}{T}_j^t = \left[\frac{SGN}{x_j} \;\; 0 \;\; \frac{SGN}{h(x_j)} \;\; 0 \;\; 0 \;\; 0\right] \qquad \text{END } A \tag{102}$$

$$\underset{\sim}{T}_j^t = \left[0 \;\; 0 \;\; 0 \;\; \frac{SGN}{x_j} \;\; 0 \;\; \frac{SGN}{h(x_j)}\right] \qquad \text{END } B \tag{103}$$

The notation SGN represents the sign of the elements of the element force vector, $\underset{\sim}{S}_j$. The parameter $h(x_j)$ is defined as (Section Modulus)

$$h(x_j) = \frac{I_j}{c_j} \tag{104}$$

where c_j is the exteme fiber distance at which the stress is of maximum magnitude. The element force matrix $\underset{\sim}{S}_j$ can be written as

$$\underset{\sim}{S}_j = \underset{\sim}{k}_j \underset{\sim}{a}_j \underset{\sim}{u} \tag{105}$$

The expression for σ_j can be written as

$$\sigma_j = \underset{\sim}{F}_j^t \underset{\sim}{u} \tag{106}$$

where the virtual load vector $\underset{\sim}{F}_j^t$ is given by

$$\underset{\sim}{F}_j^t = \underset{\sim}{T}_j^t \underset{\sim}{k}_j \underset{\sim}{a}_j \tag{107}$$

The e_{ij} (Eq 8) in the optimality condition is given by

$$e_{ij} = \frac{\delta_{ij}[\underset{\sim}{T}_j^t \overline{\underset{\sim}{S}}_j - \overline{\underset{\sim}{T}}_j^t \underset{\sim}{S}_j] - \underset{\sim}{f}_j^t [\underset{\sim}{K}_{Ai} + n_i \underset{\sim}{K}_{Bi}]\underset{\sim}{u}}{\rho_i x_i l_i} \tag{108}$$

where the new matrices $\bar{\underset{\sim}{T}}_j$ and $\bar{\underset{\sim}{S}}_j$ are defined as

$$\overline{\underset{\sim}{T}}_j^t = -\frac{\partial \underset{\sim}{T}_j^t}{\partial x_j} x_j \tag{109}$$

$$\overline{\underset{\sim}{S}}_j = (\underset{\sim}{k}_{Aj} + n_j \underset{\sim}{k}_{Bj}) \underset{\sim}{a}_j \underset{\sim}{u} \tag{110}$$

The lower case k represents the element stiffness matrix in the local coordinate system. The vector $\underset{\sim}{f}_j$ is defined in Eq 62 with the virtual load vector defined by Eq 107. δ_{ij} is the Kronecker delta.

Now the Lagrangian multiplier is given by

$$\lambda_j = -\frac{W}{\overline{Z}(\mu_{Aj} + \mu_{Bj} - \mu_j)} \tag{111}$$

The parameters μ_{Aj} and μ_{Bj} are defined as before, Eqs 69 and 70, and the virtual load vector is defined by Eq 107. The μ_j parameter is defined as

$$\mu_j = \frac{\underset{\sim}{T}_j^t \overline{\underset{\sim}{S}}_j - \overline{\underset{\sim}{T}}_j^t \underset{\sim}{S}_j}{\underset{\sim}{F}_j^t \underset{\sim}{u}} \tag{112}$$

For membrane structures $\mu_j = 0$, $\mu_{Bj} = 0$ and $\mu_{Aj} = 1$. For $n_j = n = 1$, μ_j would be nearly zero also.

The scale factor for stress constraints can be derived from Eq 30 with

$$R = \sigma_j = \underset{\sim}{F}_j^t \underset{\sim}{u} \tag{113}$$

Then Eq 30 can be written as

$$\frac{d\sigma_j}{\sigma_j} = (\Lambda - 1) \frac{\sum_{i=1}^{m} \frac{\partial \sigma_j}{\partial x_i} x_i}{\sigma_j} \tag{114}$$

After substituting Eqs 100 to 107 in 114 one can write

$$\frac{d\sigma_j}{\sigma_j} = (1 - \Lambda)(\mu_{Aj} + \mu_{Bj} - \mu_j) \tag{115}$$

The scale factor Λ can be written as

$$\Lambda = 1 - \frac{d\sigma_j}{\mu\sigma_j} = 1 - b \tag{116}$$

where

$$\mu = \mu_{Aj} + \mu_{Bj} - \mu_j \tag{117}$$

$$b = \frac{d\sigma_j}{\mu\sigma_j} \ll 1 \tag{118}$$

Now once again following the derivations of Eqs 35 to 41, the scale factor can be written as

$$\Lambda = \frac{\mu_{Aj} + \mu_{Bj} - \mu_j}{\beta + \mu_{Aj} + \mu_{Bj} - \mu_j - 1} \tag{119}$$

The nondimensional parameters μ provide valuable information on the behavior of the structure. Eq 119 is similar to the equations derived earlier for the displacement and frequency constraints.

The stress constraint case is one of the most interesting, and it is worth an examination from the algorithm implementation point of view. The optimality condition (Eq 7) states that under ideal conditions the weighted sum of the energy density (or equivalent) to weight density ratio should be the same in all the structural elements. Under very special conditons this optimality condition leads to the celebrated fully stressed design concept. The special conditions are:

a. All the elements of the structure are made of the same isotropic material.

b. The elements all have the same stress allowables, and also they are the same in tension and compression.

c. The side constraints (Eq 4) do not interfere with the fulfillment of the optimality condition.

Of course, it is a tall order to satisfy all these conditons in a reasonable (respectable) practical design problem. If any of the above conditions are violated, the stress alone cannot express the full meaning of the optimality condition. This did not deter the widespread use (or abuse) of the fully stressed design concept. However, it can be used, in an ad hoc way, to improve the designs, if it is at least treated as an inequality condition. The worst abuse is when the concept is treated as an equality condition.

It is a well known fact that the active constraints in a stress constraint problem will rapidly increase as the design approaches the optimum. If one examines the optimality condition (Eq 108), the Lagrangian multipliers (Eq 111) and the scale factor (Eq 119), it appears ominous that so many virtual load and displacement vectors have to be generated.

Even though only the forward and back substitution steps (FBS) have to be repeated for each virtual load vector, the data handling and the solution time can be quite an impediment in a large scale optimization. However, it will be shown in a later publication (AFWAL Technical Report) that this need not be the case. With some approximations it is possible to limit the number of virtual load vectors to as many as the loading conditions and be independent of the number of variables in the problem. In such a case the stress constraint problem is no more difficult than the displacement constraint problem.

e. Scale Factor and the Nondimensional Parameters

The scale factors for the membrane-bending structures derived from the first order approximation (Eqs 72, 79 and 119) are good between some reasonable limits of the target response ratio β (β^2 in case of frequency constraints). From Fig. 2 one can surmize that the error in the scale factor and the response predictions beyond the limits $0.7 \leq \beta \leq 1.4$ tend to exceed 4 to 5% for structures primarily in bending. The object is now to eliminate the limitation or extend the range of β values indefinitely without sacrificing the accuracy of the scaling predictions. This can be done very neatly by writing an interaction formula in the nondimensional parameter space μ^s. This is akin to mapping the complex membrane-bending element to a simple membrane element in the parameter space.

A linear interaction formula can be written as

$$\Lambda = \frac{\mu_{Aj}}{\bar{\mu}_{Aj}}\left(\frac{1}{\beta}\right) + \frac{\mu_{Bj}}{\bar{\mu}_{Bj}}\left(\frac{1}{\beta}\right)^{\frac{1}{\bar{n}}} \tag{120}$$

where μ_{Aj} and μ_{Bj} are the nondimensional parameters given by Eqs 69, 70. The parameters $\bar{\mu}_{Aj}$ and $\bar{\mu}_{Bj}$ are simply

$$\bar{\mu}_{Aj} = 1 \qquad \bar{\mu}_{Bj} = \bar{n} \tag{121}$$

A parabolic or other nonlinear interaction formula can be written as

$$\Lambda = \left(\frac{\mu_{Aj}}{\bar{\mu}_{Aj}}\right)^p\left(\frac{1}{\beta}\right) + \left(\frac{\mu_{Bj}}{\bar{\mu}_{Bj}}\right)^q\left(\frac{1}{\beta}\right)^{\frac{1}{\bar{n}}} \tag{122}$$

where $\bar{n}$ is the aggregate value of n, and it is defined as

$$\bar{n} = \frac{\mu_{Bj}}{\bar{\bar{\mu}}_{Bj}} \tag{123}$$

p and q are exponents that give the nonlinear interaction while $\bar{\mu}_{Bj}$ is defined as

$$\bar{\bar{\mu}}_{Bj} = \frac{\sum_{i=1}^{m} \underset{\sim}{f}_j^t \underset{\sim}{K}_{Bi} \underset{\sim}{u}}{\underset{\sim}{F}_j^t \underset{\sim}{u}} \tag{124}$$

for displacement and stress constraints on membrane-bending structures. When $n_i = n$ for all the elements, then $n = \bar{n}$.

In fact the interaction formula as defined by Eq 120 reduces the prediction errors to within one or two percent (See Fig 2) regardless of the range of β. This is the most important property of membrane elements, and it is now extended to membrane bending elements as well. Actually a nonlinear interaction formula (Eq 122) can completely eliminate the prediction error, but it takes some effort to solve for the appropriate p and q exponents. This is a fascinating result because the nondimensional parameters derived from the first order approximation (Taylor's Series) are instrumental in mapping the membrane-bending element to simple membrane element properties and eliminate the limitations inherent in such approximations.

In the case of membrane bending elements with frequency constraints the linear interaction formula can be written as

$$\Lambda = \frac{\mu_{Aj}}{\bar{\mu}_{Aj}}\left(\Lambda_{Aj}\right) + \frac{\mu_{Bj}}{\bar{\mu}_{Bj}}\left(\Lambda_{Bj}\right) \tag{125}$$

where μ_{Aj} and μ_{Bj} are defined by Eqs 80 and 81.

The parameter $\bar{\mu}_{Aj}$ and $\bar{\mu}_{Bj}$ are given by Eq 121. Now $\bar{n}$, the aggregate value of n, is once again defined by Eq 123 with the $\bar{\bar{\mu}}_{Bj}$ definition given by

$$\bar{\bar{\mu}}_{Bj} = \frac{\sum_{i=1}^{m} \underset{\sim}{\phi}_j^t \underset{\sim}{K}_{Bi} \underset{\sim}{\phi}_j}{\underset{\sim}{\phi}_j^t \underset{\sim}{K} \underset{\sim}{\phi}_j} \tag{126}$$

The parameter Λ_{Aj} is given by

$$\Lambda_{Aj} = \frac{\gamma_j \beta_j^2}{1 - \eta_j \beta_j^2} \tag{127}$$

and Λ_{Bj} is given by the solution of the transcendental equation

$$\Lambda_{Bj}^{\bar{n}} - \beta_j^2 \eta_j \Lambda_{Bj} - \gamma_j \beta_j^2 = 0 \tag{128}$$

The solution of the transcendental equation can raise some interesting questions, and they can be explored with real structural applications.

An examination of the interaction formula (Eq 125) in the light of extreme cases reveals interesting information.

CASE 1: The bending stiffness in the mode is insignificant, and it is assumed that $\mu_{Bj} \simeq 0$. Then $\mu_{Aj} = 1$ and the scale factor expression reduces to Eq 87.

CASE 2: The axial stiffness in the mode is of minor consequence and $\mu_{Aj} \simeq 0$. Then the solution of equation 128 is the scale factor. An examination of three subcases is of interest.

CASE 2a: The aggregate parameter $\bar{n} = 1$. The scale factor reduces to Eq 87.

CASE 2b: The aggregate parameter $\bar{n} > 1.0$, but the structural mass is insignificant ($\eta_j = 0$). Then the scale factor becomes

$$\Lambda = \left(\beta_j^2\right)^{\frac{1}{\bar{n}}} \tag{129}$$

CASE 2c: The nonstructural mass is insignificant ($r_j = 0$), and $\bar{n} > 1.0$. Then the scale factor becomes

$$\Lambda = \left(\beta_j^2\right)^{\frac{1}{\bar{n}-1}} \tag{130}$$

CASE 3: The aggregate $\bar{n} = 2$

$$\Lambda_{Bj} = \frac{\beta_j^2 \eta_j \pm \sqrt{\beta_j^4 \eta_j^2 + 4\gamma_j \beta_j^2}}{2} \tag{131}$$

Since the radical under the root is always greater than $\beta_j^2 \eta_j$, there is one positive and one negative root. Only the positive root is significant.

The comments made earlier about mapping the membrane-bending element to simple membrane element properties is just as valid in the case of frequency constraint problems.

SUMMARY AND CONCLUSIONS

The significance of the generalization process derived in this paper cannot be overemphasized. It breaks the barrier for the application of the optimality criteria methods to most general multidisciplinary structural optimization problems. All four important elements of the optimality criteria method are expressed as a function of a single quantity, sensitivity (i.e. gradients of the constraints and the objective functions):

Optimality Conditions	- F_1 (Sensitivity)
Lagrangian Multipliers	- F_2 (Sensitivity)
Resizing Algorithm	- F_3 (Sensitivity)
Scaling	- F_4 (Sensitivity)

A sensitivity analysis for all the disciplines that participate in aerospace structural design is readily available. For example, reference 17 [Venkayya (1985)] contains a summary of the sensitivity analysis for some of these conditions.

The basic approach of the optimality criteria method was presented earlier in a series of publications [Venkayya, Khot and Reddy (1969); Venkayya (1971); Venkayya, Khot and Berke (1973); Venkayya and Tischler (1983); Grandhi and Venkayya (1987); Canfield, Grandhi and Venkayya (1987)] by the author and his associates at the Air Force Wright Aeronautical Laboratories. However, the method was presented in the context of special design conditions and membrane structures with some indication that it could

be generalized. This created skepticism about its validity in a multidisciplinary setting. In particular, scaling, the most important element in the optimality criteria approach, was often dismissed as relevant only in the case of membrane structures and static design conditions. This paper shows that it is not the case.

A puzzling question is why the optimality criteria approach is needed and how is it different from the standard nonlinear programming approaches? The basic information needed in both these methods is not significantly different, and there appears to be a great deal of similarity. But nevertheless, how this information is used and the simplicity of the approach are the distinction.

In a standard nonlinear programming approach the search for an optimum progresses from point to point in the design space as indicated by the following equation

$$\underset{\sim}{x}^{\nu+1} = \underset{\sim}{x}^{\nu} + \alpha \underset{\sim}{D} \tag{132}$$

This equation expresses the concept of perturbation of the current design by adding (subtracting) the information derived from the sensitivity analysis in order to obtain a new design. This creates a serious drawback by searching too many points in an n-dimensional space, particularly when n is large, as in a design with large finite element assemblies. The most charitable upper limit on the number of variables that the current nonlinear programming approaches can handle is about 300, unless one professes to know (crystal ball) how to link these variables to reduce the design space. In addition, they get bogged down at every relative minimum in their path. As a result, selection of an initial design and the appropriate step size becomes a complex art and needs a disproportionate amount of attention that it does not deserve.

The search for the optimum (Eq 23) in an optimality criteria method does not progress from point to point, but instead it sweeps the design space, as indicated symbolically in Fig. 1. Of course, sweeping has little advantage without an effective scaling algorithm to estimate the location of the constraint boundary. The scaling algorithm outlined in this paper is simple and can handle all the design conditions encountered in aerospace structural design.

The weighting matrix, $\bar{A}$, as defined in the derivation of the optimality conditions not only eliminates any significant effort to obtain the Lagrangian multipliers, but also offers opportunities for extension of the method beyond structural design.

An important by product of the optimality criteria is the association of the sensitivity to some energy or equivalent in the system. Most of the analysis methods are derived from energy considerations, and as a result, the design information is naturally available from the analysis. In addition, the formulation developed extremely important design

parameters (such as $\mu, \eta, \Lambda, \beta$, etc.) which provide significant insight into the expected behavior of the structure. These parameters, together with the side constraints as defined in Eq 4, can offer a great deal of freedom in tailoring designs and their behavior at little or no extra cost.

The sweeping concept together with the scaling algorithm uncouples the number of design iterations from the number of design variables. This is a significant property that breaks the barrier of implementing formal optimization in the preliminary design of aerospace structures (in a multidisciplinary environment) using finite element models.

A proper implementation of the optimality criteria approach offers the promise that the optimal design can be completed in five to ten cycles of iteration, regardless of the number of variables and the type of constraints. This is a key requirement for transferring formal optimization to real aerospace structures design. Then the role of optimization in structural design would be more positive [Ashley (1981)].

An effective optimization as outlined in this paper offers rich dividends in the form of performance improvements as well as weight (cost) reduction in aerospace structures. The design examples shown earlier [Venkayya, Khot and Reddy (1969); Venkayya (1971); Venkayya, Khot and Berke (1973); Venkayya and Tischler (1983); Grandhi and Venkayya (1987); Canfield, Grandhi and Venkayya (1987)] attest to the conclusions drawn in this paper.

ACKNOWLEDGEMENT

This work was part of an in-house research project in the Analysis and Optimization Branch (FIBR) of the Air Force Wright Aeronautical Laboratories (AFWAL). This project was sponsored by the Air Force Office of Scientific Research (AFOSR) under the laboratory task 2302N5 of which "Multidisciplinary Optimization" is one of the components.

REFERENCES

Ashley, H., (1981): "On Making Things Best - Aeronautical Uses of Optimization," AIAA 50^{th} Anniversary, Wright Brother's Lecture, AIAA Aircraft Systems and Technology Conference, 11-13 August 1981, Dayton OH.

Canfield, R.A., Grandhi, R.V., and Venkayya, V.B. (1987): "Structural Optimization with Stiffness and Frequency Constraints," presented at AIAA/ASME/ASCE/AHS 28^{th} SDM Conference, Monterey CA, April 1987. Also appeared in J. Mechanics of Structures and Machines.

DeSalvo, G.J., and Swanson, J.A., (1985): "ANSYS - Engineering Analysis System - User's Manual," Swanson Analysis, Inc.

Dwyer, W.J., Emerton, R.K., and Ojalvo, I.U., (1971): "An Automated Procedure for the Optimization of Practical Aerospace Structures," AFFDL-TR-70-118.

Grandhi, R.V., and Venkayya, V.B., (1987): "Structural Optimization with Frequency Constraints," Proceedings of AIAA/ASME/ASCE/AHS 28^{th} Structures, Structural Dynamics and Materials Conference, Monterey CA, April, 1987. Also to appear in an AIAA Journal.

Johnson, E.H., Herendeen, D.L., and Venkayya, V.B., (1984): "A General Automated Aerospace Structural Design Tool," a paper presented at the 21^{st} Annual Meeting, Society of Engineering Science, Inc., October, 1984.

Lynch, R.W., Rogers, W.A., Braymen, G.W., and Hertz, T.J., "Aeroelastic Tailoring of Advanced Composite Structures for Military Aircraft, Volume III - Modifications and User's Guide for Procedure TSO,"

MacNeal, R.H., (1971): The Nastran Theoretical Manual, NASA-SP-221(01), April, 1971.

Mikolaj, P., (1987): "Capabilities and Special Features Concerning Structural Optimization of Spacecraft Structures," a paper presented at the 38^{th} International Astronautical Federation, Brighton, England, 16 October 1987.

Petiau, C., and Lecina, G., "Elements Finis et Optimization des Structues Aeronautiques," AGARD Conference Proceedings No. 280-The Use of Computers as a Design Tool, pp 23-1 - 23-16.

Schmit, L.A., Jr., and Miura, H., (1976): "Approximation Concepts for Efficient Structure Synthesis," NASA-CR-2552, March, 1976.

STARS, The RAE Structural Analysis and Redesign System - User Guide, Scion Ltd, July, 1984.

Venkayya, V.B., Khot, N.S., and Reddy, V.S., (1969): "Energy Distribution in an Optimum Structural Design," AFFDL-TR-68-156.

Venkayya, V.B., (1971): "Design of Optimum Structures," J. Computers and Structures-1, 265-309.

Venkayya, V.B., Khot, N.S., and Berke, L., (1973): "Application of Optimality Criteria Approaches to Automated Design of Large Practical Structures," Second Symposium on Structural Optimization, AGARD-CP-123, Milan, Italy.

Venkayya, V.B., and Tischler, V.A., (1979): "OPTSTAT - A Computer Program for the Optimal Design of Structures Subjected to Static Loads," AFFDL-TM-79-67-FBR.

Venkayya, V.B., and Tischler, V.A., (1983): "Optimization of Structures with Frequency Constraints," Computer Methods for Nonlinear Solids and Structural Mechanics, ASME AMD 54, pp 239-259.

Venkayya, V.B., (1985): "Computer-Aided Optimum Structural Design," Class Notes for a Short Course in National Cheng Kung University, Tainan, Taiwan, 19-23 August 1985.

Wilkinson, K., Markowitz, J., Lerner, E., George, D., and Batill, S.M., (1977): "FASTOP: A Flutter and Strength Optimization Program for Lifting Surface Structures," J. Aircraft, 14, pp 581-587.

4.0 DESIGN SENSITIVITY ANALYSIS

Statement of the optimization problem

Minimize an objective function (or maximize)

$$W = W(\underset{\sim}{A}) \qquad (4-1)$$

where $\underset{\sim}{A}$ is the vector of design variables

subject to the constraints

$$\psi_j(\underset{\sim}{A}) = z_j(\underset{\sim}{A}) - c_j(\underset{\sim}{A}) \leq 0 \qquad j = 1, 2, \ldots n_c \qquad (4-2)$$

where n_c represents the number of constraints.

Inequality constraints $\psi_j \leq 0 \quad (or\ \psi_j > 0)$

Equality constraints $\psi_j = 0$

Active and passive constraints

All the constraints within a prescribed distance from the boundary will be considered as active constraints.

Constraints beyond this distance are considered as passive constraints.

Only the active constraints will be considered in determining the direction of travel to the optimum. However, this active and passive constraint status will be continually updated during the course of the optimization algorithm implementation.

Gradients of the objective function

$$\frac{\partial W}{\partial \underset{\sim}{A}} = \frac{\partial W(\underset{\sim}{A})}{\partial \underset{\sim}{A}} \qquad (4-3)$$

Example: Objective Function-Weight for a truss or a frame

$$W = W(\underset{\sim}{A}) = \sum_{i=1}^{m} A_i l_i \rho_i \qquad (4-4)$$

where A_i represents the i^{th} design variable and l_i is the length or the surface parameter of the i^{th} element. ρ_i is the weight density of the material.

In the case of line elements (such as rods and beams) A_i is the cross- sectional area of the element and l_i is the length.

In the case of surface elements A_i represents the thickness of the element and l_i represents the surface area of the element. For a given configuration l_i is a fixed quantity. It only varies in shape optimization.

Then the gradient of the objective function can be written

$$\frac{\partial W}{\partial A_i} = \rho_i l_i \qquad (4-5)$$

The nice thing about the weight as the objective function is that it is a linear function in the design variables and the derivatives with respect to each design variable can be

uncoupled. In general the constraint functions are not linear but some can be formulated such that the derivatives can be uncoupled.

Gradients of the Displacement Constraints

The displacement constraints can be imposed either on individual components of the displacement vector $\underset{\sim}{u}$ or a linear combination of the components.

$$\psi_j(\underset{\sim}{A}) = z_j(\underset{\sim}{A}) - c_j(\underset{\sim}{A}) \qquad j = 1, 2, \ldots p \tag{4-6}$$

In the first case, consider a constraint on an individual displacement component

$$z_j(\underset{\sim}{A}) = u_i \tag{4-7}$$

where u_i is the i^{th} component of the displacement vector $\underset{\sim}{u}$.

Example: Tip displacement of the truss or the wing tip transverse displacement.

In the second case, consider a constraint on a linear combination of the displacement components.

$$z_j(\underset{\sim}{A}) = \alpha_1 u_i + \alpha_2 u_k + \alpha_3 u_l \tag{4-8}$$

Example: Wing Box Twist Constraint.

Going back to the constraint on the i^{th} component of displacement u_i, the gradients of the displacement constraint can be calculated in two ways which will be referred to as Method 1 and Method 2.

Method 1: Displacement constraint gradients by the virtual load method.

The constrained displacement will be defined as follows:

$$u_i \hat{=} \underset{\sim}{F}^t \underset{\sim}{u} \tag{4-9}$$

where $\underset{\sim}{F}$ and $\underset{\sim}{u}$ are $n \times 1$ vectors. The displacement vector $\underset{\sim}{u}$ is due to the applied load vector $\underset{\sim}{P}$, where $\underset{\sim}{F}$ is referred to as the virtual load vector. The elements of the virtual load vector are defined in such a way that the product on the right hand side yields the quantity u_i, the i^{th} component of the displacement vector $\underset{\sim}{u}$. By this definition all the elements of $\underset{\sim}{F}$ should be zero with the exception of the i^{th} element which will be 1.

$$F_i = 0 \quad for\ all \quad i \neq i$$

$$F_l = 1 \quad for\ all \quad l = i \tag{4-10}$$

Now the constraint ψ_j can be defined as

$$\psi_j(\underset{\sim}{A}) = \underset{\sim}{F}^t \underset{\sim}{u} - c_j(\underset{\sim}{A}) \leq 0 \tag{4-11}$$

In a purely displacement constraint problem the constant value c_j is generally independent of the design variables in which case the constraint derivative can be written as

$$\frac{\partial}{\partial A_i}\psi_j = \frac{\partial}{\partial A_i}(\underset{\sim}{F}^t)\underset{\sim}{u} + \underset{\sim}{F}^t \frac{\partial}{\partial A_i}\underset{\sim}{u} \tag{4-12}$$

or

$$\psi_{,A_i} \hat{=} \frac{\partial}{\partial A_i}\psi_j = \underset{\sim}{F}^t \frac{\partial}{\partial A_i}\underset{\sim}{u} \tag{4-13}$$

Now we define the equilibrium relations as

$$\underset{\sim}{P} = \underset{\sim}{K}\underset{\sim}{u} \tag{4-14}$$

$$\underset{\sim}{F} = \underset{\sim}{K}\underset{\sim}{f} \qquad (4-15)$$

where $\underset{\sim}{u}$ and $\underset{\sim}{f}$ are the displacement vectors resulting from the application of the force vectors $\underset{\sim}{P}$ and $\underset{\sim}{F}$ respectively. Now the displacement vector derivative can be obtained from Eq. (4-14)

$$\frac{\partial}{\partial A_i}\underset{\sim}{P} = \frac{\partial}{\partial A_i}(\underset{\sim}{K})\underset{\sim}{u} + \underset{\sim}{K}\frac{\partial}{\partial A_i}\underset{\sim}{u} \qquad (4-16)$$

Solving for the displacement vector derivative gives

$$\frac{\partial}{\partial A_i}\underset{\sim}{u} = K^{-1}\left[\frac{\partial}{\partial A_i}\underset{\sim}{P} - \frac{\partial}{\partial A_i}(\underset{\sim}{K})\underset{\sim}{u}\right] \qquad (4-17)$$

Substitution of Eq. (4-17) in Eq. (4-13) gives the expression for the constraint derivative in the form

$$\psi_{,A_i} = F^t K^{-1}\left[\frac{\partial}{\partial A_i}\underset{\sim}{P} - \frac{\partial}{\partial A_i}(\underset{\sim}{K})\underset{\sim}{u}\right] \qquad (4-18)$$

From Equation (4-15)

$$\psi_{,A_i} = f^t\left[\frac{\partial}{\partial A_i}\underset{\sim}{P} - \frac{\partial}{\partial A_i}(\underset{\sim}{K})\underset{\sim}{u}\right] \qquad (4-19)$$

In most practical cases the changes in the design variables do not significantly effect the load vector $\underset{\sim}{P}$ in which case the first term on the right vanishes.

$$\psi_{,A_i} = -f^t\frac{\partial}{\partial A_i}(\underset{\sim}{K})\underset{\sim}{u} \qquad (4-20)$$

If we recall that the stiffness matrix $\underset{\sim}{K}$ is defined as

$$\underset{\sim}{K} = \sum_{i=1}^{n_v}\underset{\sim}{a}_i^t\underset{\sim}{k}_i\underset{\sim}{a}_i \qquad (4-21)$$

where $\underset{\sim}{k}_i$ is the i^{th} element stiffness matrix and $\underset{\sim}{a}_i$ is the element to structure compatability matrix, then

$$\frac{\partial}{\partial A_i}(\underset{\sim}{K}) = \underset{\sim}{a}_i^t\,\underset{\sim}{\bar{k}}\,\underset{\sim}{a}_i = \underset{\sim}{\overline{K}}_i \qquad (4-22)$$

where $\overline{\underset{\sim}{K}}_i$ is given by

$$\overline{\underset{\sim}{K}}_i \hat{=} a_i^t \frac{\partial}{\partial A_i}(\underset{\sim}{k}_i) a_i \qquad (4-23)$$

Substituting Eq. (4-22) in Eq. (4-20) one can write the constraint gradient as

$$\psi_{,A_i} = -\underset{\sim}{f}^t \overline{\underset{\sim}{K}}_i \underset{\sim}{u} = -\underset{\sim}{f}_i^t \overline{\underset{\sim}{K}}_i \underset{\sim}{u}_i \qquad (4-24)$$

where $\underset{\sim}{f}_i$ and $\underset{\sim}{u}_i$ are the i^{th} element displacements in the global coordinate system. From Equation (4-24) one can conclude that a displacement constraint gradient represents the virtual strain energy in the element per unit value of the design variable or one might write Eq. (4-24) as

$$\psi_{,A_i} \simeq -\frac{\underset{\sim}{f}_i^t \underset{\sim}{K}_i \underset{\sim}{u}_i}{A_i} \qquad (4-25)$$

Now the procedure for displacement constraint gradients computation by the virtual load method can be outlined as follows:

1. The displacement vector $\underset{\sim}{u}$ is determined by Equation (4-14) which involves three steps.

a) Decomposition of $\underset{\sim}{K}$

$$\underset{\sim}{K} = \underset{\sim}{L}\underset{\sim}{D}\underset{\sim}{L}^t \qquad (4-26)$$

b) Forward Substitution to determine $\underset{\sim}{y}$

$$\underset{\sim}{L}\underset{\sim}{y} = \underset{\sim}{F} \qquad (4-27)$$

c) Back Substitution to determine $\underset{\sim}{y}$

$$\underset{\sim}{D}\underset{\sim}{L}^t\underset{\sim}{y} = \underset{\sim}{y} \qquad (4-28)$$

2. The displacement vector $\underset{\sim}{f}$ is determined by Equation (4-15). For this only forward and back substitutions have to be repeated since the stiffness matrix is already decomposed in the last step.

3. Now determination of the constraint gradients with respect to each variable requires simply substitution of $\underset{\sim}{f}$ and $\underset{\sim}{y}$ in Equation (4-25).

It is evident from this procedure, that to determine the constraint gradients with respect to all the variables, one decomposition and two forward and back substitutions are necessary. However, for each additional constraint only one forward and back substitution is necessary. If there are n_c constraints and n_v design variables, then the total number of constraint gradients to be evaluated is n_{cg} which is given by

$$n_{cg} = n_c \times n_v \qquad (4-29)$$

The corresponding number of decompositions is given by n_d

$$n_d = 1 \qquad (4-30)$$

The number of forward and back substitutions is n_{FBS} and is given by

$$n_{FBS} = n_c + 1 \qquad (4-31)$$

If one considers that decomposition of the stiffness matrix requires the most computational effort and FBS (Forward and Back Substitution) requires a much smaller effort, then computation of constraint gradients by the virtual load method is very appealing in view of reducing the total computational effort.

Method 2: Displacement constraint gradients by a first order Taylor Series approximation.

If $\underset{\sim}{u}$ is the displacement vector due to the applied loads, then a change in $\underset{\sim}{u}$ due to a change in the design variable vector can be written in a Taylor's series expansion.

$$\underset{\sim}{u} + d\underset{\sim}{u} = \underset{\sim}{u} + \sum_{i=1}^{n_v} \frac{\partial \underset{\sim}{u}}{\partial A_i} dA_i + \sum_{i=1}^{n_v} \sum_{j=1}^{n_v} \frac{\partial^2 \underset{\sim}{u}}{\partial A_i \partial A_j} dA_i dA_j + \cdots \qquad (4-32)$$

A first order approximation of this series can be written as

$$d\underset{\sim}{u} = \sum_{i=1}^{n_v} \frac{\partial \underset{\sim}{u}}{\partial A_i} dA_i \qquad (4-33)$$

From Equation (4-14)

$$\underset{\sim}{K} \frac{\partial \underset{\sim}{u}}{\partial A_i} = -\overline{\underset{\sim}{K}}_i \underset{\sim}{u} \qquad (4-34)$$

In the above derivation the changes in the applied load vector due to changes in the design variables are assumed to be zero.

From Equations (4-33) and (4-34) the constraint gradient can be written as

$$\psi_{,A_i} = -\underset{\sim}{K}^{-1} \overline{\underset{\sim}{K}}_i \underset{\sim}{u} \qquad (4-35)$$

The constraint gradient evaluations by a first order approximation of a Taylor Series expansion involves

$$n_d = 1$$

and

$$n_{FBS} = n_v + 1 \qquad (4-36)$$

It is evident from Equations (4-31) and (4-36) that the choice of method 1 or 2 depends on a comparison of the number of active constraints (n_c) and the number of variables (n_v).

In most preliminary design problems using finite element models the number of design variables is large compared to the number of active constraints. In such cases method 1 is definitely more advantageous. However, when the number of variables is drastically reduced by procedures such as linking, then the second method can be made competitive.

Gradients of the Stress Constraints

The stress constraints on the elements are derived from material strength considerations and/or the structural concept used in the construction. The stress-stain properties and the fatigue-fracture behavior are the important factors in material strength considerations. The element overall buckling, local buckling of components, crippling, etc. are the factors introduced by the structural concept and they would influence the values of the stress constraints. Once again the stress constraint can be represented by Equation (4-2). However, the details of the stress constraint vary with the type of element. For instance, in a simple rod (axial force member) stress constraints can be defined by simple tension and compression allowables. They can be the same or different. For a membrane plate (in a biaxial state of stress) the stress constraint definition depends on the type of failure theory used. The failure theories are as follows:

1. Maximum normal stress
2. Maximum normal strain
3. Maximum shear stress
4. Generalized energy of distortion (or Von Mises)

We will explain the stress constraint gradient in the context of the Von Mises criteria, and the other three can be treated as special cases or equivalent.

In a finite element analysis the stress in an element can be written as

$$\underset{\sim}{\sigma} = \underset{\sim}{E}\underset{\sim}{D}\underset{\sim}{\phi}\underset{\sim}{a}\underset{\sim}{u} \qquad (4-37)$$

where $\underset{\sim}{\sigma}$ is the stress vector in the element, $\underset{\sim}{E}$ is the matrix of elastic constants, $\underset{\sim}{D}$ is the differential operator and $\underset{\sim}{a}$ is the matrix that establishes compatability of the elements and the structure. For bending elements the design variable will also be part of the right hand side. For membrane elements

$$\frac{\partial \underset{\sim}{\sigma}}{\partial A} = \underset{\sim}{E}\underset{\sim}{D}\underset{\sim}{\phi}\underset{\sim}{a}\frac{\partial \underset{\sim}{u}}{\partial A} \qquad (4-38)$$

Here we made an assumption that the material selection for the element was made earlier. As an example we will consider the case when the stress vector consists of three elements $\{\sigma_x, \sigma_y, \sigma_{xy}\}^t$, where σ_x and σ_y are the normal stresses in the x and y directions respectively and σ_{xy} is the shear stress.

According to the modified energy of distortion criteria the effective stress in an element is defined as

$$\sigma_{eff}^2 = \propto_x^2 \sigma_x^2 + \propto_y^2 \sigma_y^2 - \propto_x \propto_y \sigma_x \sigma_y + \propto_{xy} \sigma_{xy}^2 \qquad (4-39)$$

where $\propto_x, \propto_y$ and $\propto_{xy}$ are functions of the allowable stresses corresponding to the σ_x, σ_y and σ_{xy} respectively. The allowable normal stresses can be different in tension and compression. For example, the tension allowable is determined by a combination of factors involving material stress-strain, fatigue and fracture properties. The compression allowable is governed by additional factors such as local buckling and crippling.

Now differentiation of Equation (4-39) with respect to the design variable gives

$$2\sigma_{eff}\frac{\partial\sigma_{eff}}{\partial A} = 2\,\alpha_x^2\,\sigma_x\frac{\partial\sigma_x}{\partial A} + 2\,\alpha_y^2\,\sigma_y\frac{\partial\sigma_y}{\partial A} - \alpha_x\alpha_y\,\frac{\partial\sigma_x}{\partial A}\sigma_y$$

$$- \alpha_x\alpha_y\,\sigma_x\frac{\partial\sigma_y}{\partial A} + 2\,\alpha_{xy}\,\sigma_{xy}\frac{\partial\sigma_{xy}}{\partial A} \qquad (4-40)$$

Now

$$\frac{\partial\sigma_{eff}}{\partial A} = \underset{\sim}{\sigma}_s^t\frac{\partial\underset{\sim}{\sigma}}{\partial A} \qquad (4-41)$$

where $\underset{\sim}{\sigma}_s^t$ is defined as

$$\underset{\sim}{\sigma}_s^t \hat{=} \begin{bmatrix}\sigma_{sx} & \sigma_{sy} & \sigma_{sxy}\end{bmatrix} \qquad (4-42)$$

and σ_{sx}, σ_{sy} and σ_{sxy} are given by

$$\sigma_{sx} = \frac{2\,\alpha_x^2\,\sigma_x - \alpha_x\alpha_y\,\sigma_y}{2\sigma_{eff}} \qquad (4-43)$$

$$\sigma_{sy} = \frac{2\,\alpha_y^2\,\sigma_y - \alpha_x\alpha_y\,\sigma_x}{2\sigma_{eff}} \qquad (4-44)$$

$$\sigma_{sxy} = \frac{2\,\alpha_{xy}^2\,\sigma_{xy}}{2\sigma_{eff}} \qquad (4-45)$$

It was tacitly assumed in Equation (4-40) that the allowable stresses are independent of the design variables. But this is not necessarily true in the case of buckling constraints. However, a modification to account for this dependency can be handled in an approximate way.

Now recalling the constraint Equation (4-2).

$$\psi(A) = \sigma_{eff} - c \leq 0 \qquad (4-46)$$

$$\frac{\partial\psi}{\partial A} = \underset{\sim}{\sigma}_s^t\frac{\partial\underset{\sim}{\sigma}}{\partial A} \qquad (4-47)$$

Equations (4-38) and (4-47) give the stress constraint gradient.

Now the stress constraint gradient can be determined by either method 1 or method 2.

Method 1: Stress Constraint Gradients by the Virtual Load Method

The stress in an element can be expressed as

$$\underset{\sim}{\sigma} \hat{=} \underset{\sim}{F}^t \underset{\sim}{u} \qquad (4-48)$$

where $\underset{\sim}{F}$ is the virtual load vector on the structure and $\underset{\sim}{u}$ is the displacement vector due to the applied loads. Now a comparison of Equations (4-37) and (4-48) yields the definition of the virtual load vector as follows:

$$\underset{\sim}{F}^t = \underset{\sim}{E}\underset{\sim}{D}\underset{\sim}{\phi}\underset{\sim}{a} \qquad (4-49)$$

Now differentiation of Equation (4-48) with respect to the design variable gives

$$\frac{\partial \underset{\sim}{\sigma}}{\partial A_i} = \frac{\partial \underset{\sim}{F}^t}{\partial A_i}\underset{\sim}{u} + \underset{\sim}{F}^t \frac{\partial \underset{\sim}{u}}{\partial A_i} \qquad (4-50)$$

The first term in Equation (4-50) is generally zero except in the case of bending elements. Then Equations (4-17) and (4-50) can be written as

$$\frac{\partial \underset{\sim}{\sigma}}{\partial A_i} = -\underset{\sim}{F}^t \underset{\sim}{K}^{-1} \frac{\partial \underset{\sim}{K}}{\partial A_i}\underset{\sim}{u} \qquad (4-51)$$

Now the stress constraint gradient can be written as

$$\frac{\partial \underset{\sim}{\sigma}}{\partial A_i} = -\underset{\sim}{f}_i^t \underset{\sim}{\bar{K}}_i \underset{\sim}{u}_i \qquad (4-52)$$

Equation (4-52) represents the virtual strain energy in the elements per unit value of the design variable due to the virtual load and the actual applied load. It should be noted that

Equations (4-24) and (4-52) are exactly similar in form, but the definition of the applied virtual load is distinctly different.

Now the stress constraint gradient with respect to the design variable can be written from Equations (4-47) and (4-52) as

$$\psi_{,A_i} = -\underset{\sim}{\sigma}_s^t \underset{\sim}{f}_i^t \overline{\underset{\sim}{K}}_i \underset{\sim}{u}_i \qquad (4-53)$$

Method 2: Stress Constraint Gradient by a First Order Taylor Series Approximation

The stress constraint gradient by a first order approximation of a Taylor Series can be written combining Equations (4-34), (4-38) and (4-47).

$$\psi_{,A_i} = -\underset{\sim}{\sigma}_s^t \underset{\sim}{E} \underset{\sim}{D} \underset{\sim}{\phi}_a \underset{\sim}{K}^{-1} \overline{\underset{\sim}{K}}_i \underset{\sim}{u}_i \qquad (4-54)$$

The computational effort required for the two methods is similar to that presented under the displacement constraint gradients.

Gradients of the Eigenvalues and Eigenvectors

The generalized linear eigenvalue problem can be written as

$$\underset{\sim}{A}\underset{\sim}{X} = \lambda \underset{\sim}{B}\underset{\sim}{X} \qquad (4-55)$$

Most free vibration and buckling problems can be represented by Equation (4-55). For vibration problems the $\underset{\sim}{A}$ and $\underset{\sim}{B}$ matrices represent the stiffness and mass matrices respectively. In all second order differential equation representations of free-vibration problems, $\underset{\sim}{B}$ is generally symmetric and positive definite and $\underset{\sim}{A}$ is also symmetric and at least positive semidefinite. In buckling problems the $\underset{\sim}{A}$ and $\underset{\sim}{B}$ matrices represent the linear and

geometric stiffness matrices respectively. In such problems both $\underset{\sim}{A}$ and $\underset{\sim}{B}$ are symmetric positive definite matrices.

For a given eigenvalue (k) and the corresponding eigenvector $\underset{\sim}{X}_k$ Equation (4-55) can be written as

$$[\underset{\sim}{A} - \lambda_k \underset{\sim}{B}]\,\underset{\sim}{X}_k = 0 \qquad (4-56)$$

And the normalization equation can be written as

$$\underset{\sim}{X}_k^t \underset{\sim}{B} \underset{\sim}{X}_k = 1 \qquad (4-57)$$

Differentiating Equations (4-56) and (4-57) with respect to the i^{th} variable results in the following equations

$$(\underset{\sim}{A}_{,i} - \lambda_{k,i}\underset{\sim}{B} - \lambda_k \underset{\sim}{B}_{,i})\underset{\sim}{X}_k + (\underset{\sim}{A} - \lambda_k \underset{\sim}{B})\underset{\sim}{X}_{k,i} = 0 \qquad (4-58)$$

$$\underset{\sim}{X}_{k,i}\underset{\sim}{M}\underset{\sim}{X}_k + \underset{\sim}{X}_k^t \underset{\sim}{M}_{,i}\underset{\sim}{X}_k + \underset{\sim}{X}_k^t \underset{\sim}{M}\underset{\sim}{X}_{k,i} = 0 \qquad (4-59)$$

These equations can be rewritten in the following form

$$(\underset{\sim}{A} - \lambda_k \underset{\sim}{B})\underset{\sim}{X}_{k,i} - \lambda_{k,i}\underset{\sim}{B}\underset{\sim}{X}_k = -(\underset{\sim}{A}_{,i} - \lambda_k \underset{\sim}{B}_{,i})\underset{\sim}{X}_k \qquad (4-60)$$

$$\underset{\sim}{X}_k^t \underset{\sim}{M}\underset{\sim}{X}_{k,i} = -\frac{1}{2}\underset{\sim}{X}_k^t \underset{\sim}{M}_{,i}\underset{\sim}{X}_k \qquad (4-61)$$

In matrix notation the above relations can be written as

$$\begin{bmatrix} (\underset{\sim}{A} - \lambda_k \underset{\sim}{B}) & -\underset{\sim}{B}\underset{\sim}{X}_k \\ \underset{\sim}{X}_k^t \underset{\sim}{B} & 0 \end{bmatrix} \begin{bmatrix} \underset{\sim}{X}_{k,i} \\ \lambda_{k,i} \end{bmatrix} = \begin{bmatrix} -(\underset{\sim}{A}_{,i} - \lambda_k \underset{\sim}{B}_{,i})\underset{\sim}{X}_k \\ -\frac{1}{2}\underset{\sim}{X}_k^t \underset{\sim}{M}_{,i}\underset{\sim}{X}_k \end{bmatrix} \qquad (4-62)$$

Now both the eigenvector derivative $\underset{\sim}{X}_{k,i}$ and the eigenvalue derivative $\lambda_{k,i}$ can be obtained (theoretically) by the solution of the linear Equations (4-62). However, the coefficient

matrix on the right hand side is singular and the solution is not easy to obtain. One approach is to find a proper pivoting procedure and solve the system of equations directly. However, pivoting over all $n+1$ equations can destroy the sparseness and/or symmetric and unsymmetric bandedness properties of the $\underset{\sim}{A}$ and $\underset{\sim}{B}$ matrices.

An alternate procedure is to solve the eigenvalue and eigenvector derivatives independently. To obtain the eigenvalue derivative first pre-multiply equation (4-60) by $\underset{\sim}{X}_k^t$

$$\underset{\sim}{X}_k^t(\underset{\sim}{A}-\lambda_k\underset{\sim}{B})\underset{\sim}{X}_{k,i}-\lambda_{k,i}\underset{\sim}{X}_k^t\underset{\sim}{B}\underset{\sim}{X}_k=-\underset{\sim}{X}_k^t\underset{\sim}{A}_{,i}\underset{\sim}{X}_k+\lambda_k\underset{\sim}{X}_k^t\underset{\sim}{B}_{,i}\underset{\sim}{X}_k \qquad (4-63)$$

Then the eigenvector derivative can be written from Equations (4-56) and (4-57)

$$\lambda_{k,i}=\underset{\sim}{X}_k^t\underset{\sim}{A}_{,i}\underset{\sim}{X}_k-\lambda_k\underset{\sim}{X}_k^t\underset{\sim}{B}_{,i}\underset{\sim}{X}_k \qquad (4-64)$$

Now to determine the eigenvector derivative the following procedure is adopted:

Equation (4-60) can be written as

$$(\underset{\sim}{A}-\lambda_k\underset{\sim}{B})\underset{\sim}{X}_{k,i}=\lambda_{k,i}\underset{\sim}{B}\underset{\sim}{X}_k-\underset{\sim}{A}_{,i}\underset{\sim}{X}_k+\lambda_k\underset{\sim}{B}_{,i}\underset{\sim}{X}_k \qquad (4-65)$$

Since the coefficient matrix on the left hand side is singular, a direct solution of Equation (4-65) is not possible. Instead we will assume the solution of Equation (4-65) as follows:

$$\underset{\sim}{X}_{k,i}=\underset{\sim}{V}_k+\propto\underset{\sim}{X}_k \qquad (4-66)$$

where $\underset{\sim}{V}_k$ is the particular solution of Equation (4-65) and is obtained by fixing one component of the eigenvector derivative and solving for the remaining components. This is done by identifying the largest component (absolute value) of the eigenvector and fixing

the corresponding component of the eigenvector derivative to zero. Now solve for the remaining components of $\underset{\sim}{V}_k$ from Equation (4-65). Now to obtain α, substitute Equation (4-66) in Equation (4-61) and solve for α.

$$\alpha = -\underset{\sim}{X}_j^t \underset{\sim}{B} \underset{\sim}{Y}_j - \frac{1}{2} \underset{\sim}{X}_j^t \underset{\sim}{M}_{,i} \underset{\sim}{X}_j \qquad (4-67)$$

In the case of vibration problems the matrices $\underset{\sim}{A}$ and $\underset{\sim}{B}$ are the stiffness and mass matrices respectively, and λ is the square of the circular frequency of vibration.

$$\underset{\sim}{A} = \underset{\sim}{K} \qquad \underset{\sim}{B} = \underset{\sim}{M} \qquad \lambda = \omega^2 \qquad (4-68)$$

For buckling problems the two matrices $\underset{\sim}{A}$ and $\underset{\sim}{B}$ are the linear stiffness, $\underset{\sim}{K}$, and the geometric stiffness matrix, $\underset{\sim}{K}_g$, respectively and λ is the buckling load factor.

$$\underset{\sim}{A} = \underset{\sim}{K} \qquad \underset{\sim}{B} = -\underset{\sim}{K}_g \qquad \lambda = \lambda(p) \qquad (4-69)$$

Eigenvalue and Eigenvector Derivatives of a Non-symmetric Matrix

Consider the eigenvalue Problem

$$(\underset{\sim}{A} - \lambda_k \underset{\sim}{I}) \underset{\sim}{X}_k = 0 \qquad (4-70)$$

where $\underset{\sim}{A}$ is a non-symmetric matrix, and λ_k and $\underset{\sim}{X}_k$ are the i^{th} eigenvalue and right eigenvector respectively. The left eigenvector $\underset{\sim}{Y}_k$ associated with λ_k is defined by the equation

$$(\underset{\sim}{A}^t - \lambda_k \underset{\sim}{I}) \underset{\sim}{Y}_k = 0 \qquad (4-71)$$

The left and right eigenvectors are equal when $\underset{\sim}{A}$ is symmetric ($\underset{\sim}{A} = \underset{\sim}{A}^t$).

The n eigenvalues of the $\underset{\sim}{A}$ matrix are determined by solution of an n^{th} order polynomial defined by

$$det\ [\underset{\sim}{A} - \lambda \underset{\sim}{I}] = 0 \qquad (4-72)$$

If the n eigenvalues are distinct, the n independent right eigenvectors exist and are biorthonormal to a set of n independent left eigenvectors,

$$\underset{\sim}{Y}_k^t \underset{\sim}{X}_l = \delta_{kl} \qquad (4-73)$$

where δ_{kl} is the Kronecker delta. Both the eigenvalues and the eigenvectors may be complex.

Differentiating Equation (4-70) with respect to the design variable

$$(\underset{\sim}{A} - \lambda_k \underset{\sim}{I})\underset{\sim}{X}_{k,i} = -[\underset{\sim}{A}_{,i} - \lambda_{k,i}\underset{\sim}{I}]\underset{\sim}{X}_k \qquad (4-74)$$

and premultiplying (4-74) by $\underset{\sim}{Y}_k^t$ gives

$$\lambda_{k,i} = \underset{\sim}{Y}_k^t \underset{\sim}{A}_{,i} \underset{\sim}{X}_k \qquad (4-75)$$

Now substituting Equation (4-75) in (4-74) gives

$$(\underset{\sim}{A} - \lambda_k \underset{\sim}{I})\underset{\sim}{X}_{k,i} = \underset{\sim}{X}_k(\underset{\sim}{Y}_k^t \underset{\sim}{A}_{,i} \underset{\sim}{X}_k) - \underset{\sim}{A}_{,i}\underset{\sim}{X}_k \qquad (4-76)$$

However, the above system of equations cannot be solved because the matrix $\underset{\sim}{A} - \lambda_k \underset{\sim}{I}$ is singular and is of rank $n-1$.

Now let us write the right eigenvector derivative as before

$$\underset{\sim}{X}_{k,i} = \underset{\sim}{V}_k + \propto \underset{\sim}{X}_k \qquad (4-77)$$

Let us also define the norm condition as follows:

$$\underset{\sim}{X}_k^{c^t} \underset{\sim}{M} \underset{\sim}{X}_k = 1 \qquad (4-78)$$

where the vector $\underset{\sim}{X}_k^c$ is the complex conjugate of $\underset{\sim}{X}_k$.

Now differentiation of Equation (4-78) with respect to the i^{th} design variable gives

$$2\mathrm{Re}(\underset{\sim}{X}_k^{c^t} \underset{\sim}{M} \underset{\sim}{X}_{k,i}) + \underset{\sim}{X}_k^{c^t} \underset{\sim}{M}_{,i} \underset{\sim}{X}_k = 0 \qquad (4-79)$$

Now substitution of Equation (4-77) in (4-79) gives as

$$\propto = -\mathrm{Re}(\underset{\sim}{X}_k^{c^t} \underset{\sim}{M} \underset{\sim}{V}_k) - \frac{1}{2} \underset{\sim}{X}_k^{c^t} \underset{\sim}{M}_{,i} \underset{\sim}{X}_k \qquad (4-80)$$

Now the procedure for determining the eigenvalue and eigenvector derivative can be outlined as follows:

1. Determine the right and left eigenvectors corresponding to the eigenvalue λ_k.

2. Then determine the eigenvalue derivative by

$$\lambda_{k,i} = \underset{\sim}{Y}_k^t \underset{\sim}{A}_{,i} \underset{\sim}{X}_k \qquad (4-81)$$

3. Then write the eigenvector derivative as

$$\underset{\sim}{X}_{k,i} = \underset{\sim}{V}_k + \propto \underset{\sim}{X}_k \qquad (4-82)$$

4. Now determine the particular solution V_k by solving Equation (4-76) after eliminating the pivotal row and column from the homogeneous system. The pivotal row and column are chosen by selecting the largest $|x_l| \cdot |y_l|$ and setting the corresponding component of the eigenvector derivative to zero. Now the remaining $(n-1)$ equations can be solved for the $(n-1)$ components of the eigenvector derivative.

5. The value of $\propto$ is determined by Equation (4-67).

6. The left eigenvector derivative can be written as

$$\underset{\sim}{Y}_{k,i} = \underset{\sim}{W}_k + \beta \underset{\sim}{Y}_k \qquad (4-83)$$

The procedure for determining $\underset{\sim}{W}_k$ is similar to that outlined for $\underset{\sim}{V}_k$. The constant β can be shown to be given by

$$\beta = -(\underset{\sim}{V}_k^t \underset{\sim}{Y}_k + \underset{\sim}{W}_k^t \underset{\sim}{X}_k + \propto) \qquad (4-84)$$

Flutter Velocity Derivatives

The most general way of expressing a flutter condition is as follows:

$$\left(-\omega^2 \underset{\sim}{M} + j\omega \underset{\sim}{C} + \underset{\sim}{K} - \frac{\rho V^2}{2}\left[\underset{\sim}{a}\left(\frac{\omega b}{V}, m\right)\right]\right) \underset{\sim}{q} = 0 \qquad (4-85)$$

where $\underset{\sim}{M}, \underset{\sim}{C}$ and $\underset{\sim}{K}$ are the mass, damping and stiffness matrices respectively and $j = \sqrt{-1}$ is an imaginary number. ω is the circular frequency of vibration at the flutter condition or simply the flutter frequency. V is the free stream velocity or the flutter speed, b is the reference aerodynamic chord, m is the Mach number, $m = V/a$, and a is the speed of sound corresponding to the altitude density ρ. $\underset{\sim}{a}$ represents the aerodynamic matrix which is a function of the reduced frequency and the Mach number, and $\underset{\sim}{q}$ is the vector of generalized coordinates. The reduced frequency k is defined as

$$k = \frac{\omega b}{V} \qquad (4-86)$$

The matrices $\underset{\sim}{M}$, $\underset{\sim}{C}$, and $\underset{\sim}{K}$ may be complex as they include the frequency response functions of the servos and controls. Also, the aerodynamic matrix $\underset{\sim}{a}$ is complex and depends on the reduced frequency and the Mach number in a transcendental form.

To the homogeneous flutter Equation (4-85) we will add the normalizing condition

$$\underset{\sim}{q}^t \underset{\sim}{W} \underset{\sim}{q} = 1 \qquad (4-87)$$

Equations (4-85) and (4-87) are a nonlinear system of equations in $2n+2$ unknowns, ω , V, $R_e(\underset{\sim}{q})$ and $I_m(\underset{\sim}{q})$, where Re is the real part and I_m the imaginary part of the complex number.

Equation (4-85) can be rewritten as

$$\underset{\sim}{F}\underset{\sim}{q} = 0 \qquad (4-88)$$

where $\underset{\sim}{F}$ is

$$\underset{\sim}{F} = -\omega^2 \underset{\sim}{M} + j\omega \underset{\sim}{C} + \underset{\sim}{K} - \frac{\rho V^2}{2}\left[\underset{\sim}{a}\left(\frac{\omega b}{V}, m\right)\right] \qquad (4-89)$$

Differentiating Equation (4-88) with respect to the design variable i gives

$$\underset{\sim}{F}_{,i}\underset{\sim}{q} + \underset{\sim}{F}\underset{\sim}{q}_{,i} = 0 \qquad (4-90)$$

$$\underset{\sim}{F}_{,i} = -2\omega\omega_{,i}\underset{\sim}{M} - \omega^2 \underset{\sim}{M}_{,i} + j\omega_{,i}\underset{\sim}{C} + j\omega \underset{\sim}{C}_{,i} + \underset{\sim}{K}_{,i} - \rho V \underset{\sim}{a} V_{,i}$$
$$- \frac{\rho V^2}{2}\underset{\sim}{a}_{,k} k_{,i} - \frac{\rho V^2}{2}\underset{\sim}{a}_{,m} m_{,i} - \frac{\rho V^2}{2}\underset{\sim}{a}_{,i}$$

Now $\underset{\sim}{a}_{,k}$ is the derivative with respect to the reduced frequency and $k_{,i}$ is given by

$$k_{,i} = b\frac{(\omega_{,i}V - \omega V_{,i})}{V^2} \qquad (4-91)$$

Now $\underset{\sim}{F}_{,i}$ can be written as

$$\underset{\sim}{F}_{,i} = -2\omega \underset{\sim}{M}\omega_{,i} - \omega^2 \underset{\sim}{M}_{,i} + j\omega_{,i}\underset{\sim}{C} + j\omega \underset{\sim}{C}_{,i} + \underset{\sim}{K}_{,i} - \rho V \underset{\sim}{a} V_{,i}$$

$$- \frac{\rho V^2}{2}\underset{\sim}{a}_{,k}\frac{b(\omega_{,i}V - \omega V_{,i})}{V^2} - \frac{\rho V^2}{2}\underset{\sim}{a}_{,m}\frac{V_{,i}}{a} - \frac{\rho V^2}{2}\underset{\sim}{a}_{,i} \qquad (4-92)$$

If we define new quantities $\overline{W}$, $\overline{Z}$ and $\bar{r}$ as

$$\overline{W} = \textit{coefficient of } \omega_{,i} = -2\omega \underset{\sim}{M} + j\underset{\sim}{C} - \frac{\rho V b}{2} \underset{\sim}{a}_{,k}$$

$$\overline{Z} = \textit{coefficient of } V_{,i} = -\frac{\rho V^2}{2a} \underset{\sim}{a}_{,m} - \rho V \underset{\sim}{a} + \frac{\rho \omega b}{2} \underset{\sim}{a}_{,k} \qquad (4-93)$$

$$\bar{r} = \textit{remaining terms} = \omega^2 \underset{\sim}{M}_{,i} - j\omega \underset{\sim}{C}_{,i} - \underset{\sim}{K}_{,i} - \frac{\rho V^2}{2} \underset{\sim}{a}_{,i}$$

then Equation (4-92) can be written as

$$F_{,i} = \overline{W}\omega_{,i} + \overline{Z}V_{,i} - \bar{r} \qquad (4-94)$$

Now Equation (4-90) can be written as

$$\underset{\sim}{F}\underset{\sim}{q}_{,i} + \underset{\sim}{F}_{,i}\underset{\sim}{q} = 0 \qquad (4-95)$$

Substituting Equation (4-94) in (4-95) gives

$$\underset{\sim}{F}\underset{\sim}{q}_{,i} + \overline{\underset{\sim}{W}}\underset{\sim}{q}\omega_{,i} + \overline{\underset{\sim}{Z}}\underset{\sim}{q}V_{,i} - \bar{r}\underset{\sim}{q} = 0 \qquad (4-96)$$

or

$$\underset{\sim}{F}\underset{\sim}{q}_{,i} + \underset{\sim}{W}\omega_{,i} + \underset{\sim}{Z}V_{,i} - \underset{\sim}{r} = 0 \qquad (4-97)$$

Equation (4-97) is a complex equation and the real and imaginary parts can be written as separate equations

$$(\underset{\sim}{F}_R + j\underset{\sim}{F}_I)(\underset{\sim}{q}_{R,i} + j\underset{\sim}{q}_{I,i}) + (\underset{\sim}{W}_R + j\underset{\sim}{W}_I)\omega_{,i} + (\underset{\sim}{Z}_R + j\underset{\sim}{Z}_I)V_{,i} = \underset{\sim}{r}_R + j\underset{\sim}{r}_I \qquad (4-98)$$

$$\underset{\sim}{F}_R\underset{\sim}{q}_{R,i} - \underset{\sim}{F}_I\underset{\sim}{q}_{I,i} + j(\underset{\sim}{F}_R\underset{\sim}{q}_{I,i} + \underset{\sim}{F}_I\underset{\sim}{q}_{R,i}) + (\underset{\sim}{W}_R + j\underset{\sim}{W}_I)\omega_{,i} + (\underset{\sim}{Z}_R + j\underset{\sim}{Z}_I)V_{,i} = \underset{\sim}{r}_R + j\underset{\sim}{r}_I \qquad (4-99)$$

$$\underset{\sim}{F}_R \underset{\sim}{q}_{R,i} - \underset{\sim}{F}_I \underset{\sim}{q}_{I,i} + \underset{\sim}{W}_R \omega_{,i} + \underset{\sim}{Z}_R V_{,i} = \underset{\sim}{r}_R$$

$$\underset{\sim}{F}_I \underset{\sim}{q}_{R,i} + \underset{\sim}{F}_R \underset{\sim}{q}_{I,i} + \underset{\sim}{W}_I \omega_{,i} + \underset{\sim}{Z}_I V_{,i} = \underset{\sim}{r}_I \qquad (4-100)$$

Now differentiating Equation (4-87) with respect to the design variable gives

$$\underset{\sim}{q}_{,i} \underset{\sim}{W} \underset{\sim}{q} + \underset{\sim}{q}^t \underset{\sim}{W} \underset{\sim}{q}_{,i} = 0 \qquad (4-101)$$

The weighting matrix is assumed to be independent of the design variable. Equation (4-101) can also be written as

$$2\underset{\sim}{q}^t \underset{\sim}{W} \underset{\sim}{q}_{,i} = 0 \qquad (4-102)$$

$$(\underset{\sim}{q}_R + j\underset{\sim}{q}_I)^t \underset{\sim}{W} (\underset{\sim}{q}_{R,i} + j\underset{\sim}{q}_{I,i}) = 0 \qquad (4-103)$$

$$\underset{\sim}{q}_R^t \underset{\sim}{W} \underset{\sim}{q}_{R,i} - \underset{\sim}{q}_I^t \underset{\sim}{W} \underset{\sim}{q}_{I,i} + j(\underset{\sim}{q}_I^t \underset{\sim}{W} \underset{\sim}{q}_{R,i} + \underset{\sim}{q}_R^t \underset{\sim}{W} \underset{\sim}{q}_{I,i}) = 0 \qquad (4-104)$$

Now equating the real and imaginary parts separately to zero gives

$$\underset{\sim}{q}_R^t \underset{\sim}{W} \underset{\sim}{q}_{R,i} - \underset{\sim}{q}_I^t \underset{\sim}{W} \underset{\sim}{q}_{I,i} = 0$$

$$\underset{\sim}{q}_I^t \underset{\sim}{W} \underset{\sim}{q}_{R,i} + \underset{\sim}{q}_R^t \underset{\sim}{W} \underset{\sim}{q}_{I,i} = 0 \qquad (4-105)$$

Combining Equations (4-100) and (4-105) we can write

$$\begin{pmatrix} \underset{\sim}{F}_R & -\underset{\sim}{F}_I & \underset{\sim}{W}_R & \underset{\sim}{Z}_R \\ \underset{\sim}{F}_I & \underset{\sim}{F}_R & \underset{\sim}{W}_I & \underset{\sim}{Z}_I \\ \underset{\sim}{q}_R^t \underset{\sim}{W} & -\underset{\sim}{q}_I^t \underset{\sim}{W} & 0 & 0 \\ \underset{\sim}{q}_I^t \underset{\sim}{W} & \underset{\sim}{q}_r^t \underset{\sim}{W} & 0 & 0 \end{pmatrix} \begin{pmatrix} \underset{\sim}{q}_{R,i} \\ \underset{\sim}{q}_{I,i} \\ \omega_{,i} \\ V_{,i} \end{pmatrix} = \begin{pmatrix} \underset{\sim}{r}_R \\ \underset{\sim}{r}_I \\ 0 \\ 0 \end{pmatrix} \qquad (4-106)$$

Equation (4-106) can be solved directly with real arithmetic, since it is, in general, not a large system of equations. Generally only $\omega_{,i}$ and $V_{,i}$ are needed and they can be determined by using the expressions given below in

$$\lambda_{,i} = \frac{\left[\underset{\sim}{p}^t(\underset{\sim}{K}_{,i} - \lambda \underset{\sim}{M}_{,i})\underset{\sim}{q} - \lambda \underset{\sim}{p}^t \underset{\sim}{A}_{,k} \underset{\sim}{q} \underset{\sim}{K}_{,i}\right]}{\underset{\sim}{p}^t(\underset{\sim}{M} + \underset{\sim}{A})\underset{\sim}{q}} \qquad (4-107)$$

where the following modifications are assumed for the flutter equation

$$[-\lambda \underset{\sim}{M} + \underset{\sim}{K} + \underset{\sim}{A}]\,\underset{\sim}{q} = 0 \qquad (4-108)$$

The associated left eigenvector $\underset{\sim}{p}$ is defined as the solution of

$$\underset{\sim}{p}^t\,[-\lambda \underset{\sim}{M} + \underset{\sim}{K} + \underset{\sim}{A}] = 0 \qquad (4-109)$$

Then the flutter velocity derivative can be written as

$$V_{,i} = -\frac{b\omega}{k^2} K_{,i} - \frac{b\omega^3}{2k} \bar{\lambda}_{,i} \qquad (4-110)$$

where $\lambda = \omega^2$ and $\bar{\lambda} = \frac{1}{\omega^2}$

If Equations (4-85) and (4-87) are used as a nonlinear system for evaluating the flutter solution and it is evaluated by the Newton-Raphson method, then the iterative solution consists of the solution of a system of linear equations with the same coefficient matrix but with different right-hand sides. There fore, evaluation of the flutter derivatives simply requires another solution with $\underset{\sim}{r}$ as the known right-hand side.

Aeroelastic Divergence

The aeroelastic divergence equation can be obtained from Equation (4-85) by setting $\omega = 0$

$$\left(-\frac{\rho V^2}{2}[\underset{\sim}{a}(m)] + \underset{\sim}{K}\right)\underset{\sim}{q} = 0 \qquad (4-111)$$

where $\underset{\sim}{a}$ and $\underset{\sim}{K}$ are real matrices.

Equation (4-111) represents a linear eigenvalue problem and $\frac{\rho V^2}{2}$ is the eigenvalue when the Mach number, m, is held constant. In such a case the eigenvalue problem has to be solved for different ρ values in order to cover all the altitudes of the flight envelope.

On the other hand if the critical value of V has to be matched for a given altitude, it will become a nonlinear eigenvalue problem, since the unknown appears in the aerodynamic matrix through m.

The first derivatives of $\underset{\sim}{q}$ and V can be obtained from

$$\left(\begin{array}{c|c} \underset{\sim}{K} - \frac{\rho V^2}{2}\underset{\sim}{a} & -\left(\rho V \underset{\sim}{a} + \frac{\rho V^2}{2a}\underset{\sim}{a}_{,m}\right) \\ \hline \underset{\sim}{q}^t \underset{\sim}{W} & 0 \end{array}\right)\left(\begin{array}{c} \underset{\sim}{q}_{,i} \\ \hline V_{,i} \end{array}\right) = \left(\begin{array}{c} \left(\frac{\rho V^2}{2}\underset{\sim}{a}_{,i} - \underset{\sim}{K}_{,i}\right)\underset{\sim}{q} \\ \hline 0 \end{array}\right) \qquad (4-112)$$

REFERENCES

1. Venkayya, V. B., "Design of Optimum Structures" Computer and Structures - Volume I, pp. 265-309, Pergammon Press.

2. Venkayya, V. B., Khot, N. S., Berke, L., "Application of Optimality Criteria Approaches to Automated Design of Large Practical Structures", Second Symposium on Structural Optimization, AGARD-CP-123, Milan, Italy.

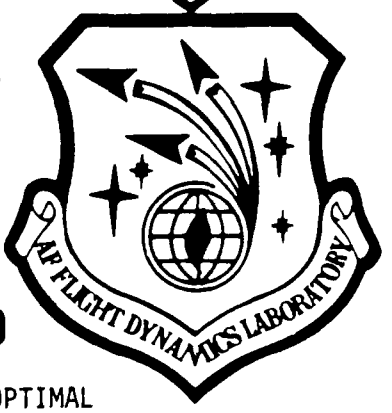
3. Adelman, H., Haftka, R. T., "Sensitivity Analysis for Discrete Structural Systems", AIAA Journal Vol 24, pp. 823-832, 1986.

4. Nelson, R. B., "Simplified Calculation of Eigenvector Derivatives" AIAA Journal, Volume 14, No 9, pp. 1201-1205, 1976.

5. Rudisill, C. S., Bhatia, K. G., "Optimization of Complex Structures to Satisfy Flutter Requirements", AIAA Journal, Volume 9, No 8, August 1971, pp. 1487-1491.

6. Arora, J. S., Haag, E. J., "Methods of Design Sensitivity Analysis in Structural Optimization", AIAA Journal, Volume 17, No 9, pp. 970-974, 1979.

TM-FBR-79-67



AIR FORCE FLIGHT DYNAMICS LABORATORY
DIRECTOR OF SCIENCE & TECHNOLOGY
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE OHIO

OPTSTAT - A COMPUTER PROGRAM FOR THE OPTIMAL DESIGN OF STRUCTURES SUBJECTED TO STATIC LOADS

VIPPERLA B. VENKAYYA

VICTORIA A. TISCHLER

JUNE 1979

TECHNICAL MEMORANDUM FBR-79-67

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligations whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed and is approved for publication.

Hartley M. Caldwell

HARTLEY M. CALDWELL, III, Capt, USAF
Chief, Analysis and Optimization Br.
Structures and Dynamics Division

FOREWORD

"OPTSTAT" is a structural optimization program with membrane elements and is intended for the design of structures subjected to static loads. The program has been in use in various forms for the past ten years. It was developed primarily for in-house research in structural optimization. It is an incore program with four membrane elements.

The program was developed under Task 240102, "Design and Analysis Methods for Aerospace Vehicles", Work Unit 24010208, "Automated Design and Analysis Methods". Captain Hartley M. Caldwell, III is the Task Engineer. The manuscript was originally released by the Authors in June, 1979.

TABLE OF CONTENTS

SECTION	TITLE	PAGE
1	INTRODUCTION	
2	ANALYSIS	
3	FINITE ELEMENTS	
4	OPTIMIZATION	
	4.1 OPTIMALITY CRITERIA FOR STIFFNESS CONSTRAINTS	
	4.2 ITERATIVE ALGORITHMS	
5	PROGRAM ORGANIZATION	
6	DESCRIPTION OF SUBROUTINES	
7	INPUT INSTRUCTIONS	
8	OUTPUT DESCRIPTION	
9	SAMPLE PROBLEMS	
APPENDIX A:	ESTIMATION OF CORE REQUIREMENTS	
APPENDIX B:	LISTING OF THE PROGRAM	
APPENDIX C:	LISTING OF THE SAMPLE DATA	
APPENDIX D:	RESULTS OF SAMPLE PROBLEMS	

ABSTRACT

This report contains documentation for the program "OPTSTAT". The program is intended for the optimization of aerospace structures modeled with membrane elements and subjected to static loads. The weight of the structure is the merit function in optimization. The constraints are on stresses, displacements and sizes of the elements. The program library consists of a bar, a membrane triangle, a membrane quadrilateral and a shear panel. The bar and shear panel can only be used with materials having isotropic or equivalent isotropic properties. The triangle and quadrilateral can be used with isotropic, orthotropic or layered composite materials.

The equations of finite element analysis, element formulations, description of the optimization algorithms, program organization and subroutine descriptions provide a comprehensive theoretical background for the program. The input and output descriptions together with the sample problem and the results should provide adequate information for the use of this program.

1. INTRODUCTION

"OPTSTAT" is an acronym for the program OPTimization of Structures for STATic loads. The program was primarily intended for in-house research studies in structural optimization. Various versions of this program have been used by the authors in the development of structural optimization algorithms over the past ten years. The results obtained from the earlier versions were published in a number of reports and papers[1-6]. Efficiency and conciseness were the driving factors in the evolution of the program. Since it was not intended to be a production program, no particular attention was paid to user convenience.

This program was distributed earlier with makeshift input and output instructions. It was used for four years (1973 - 1976) as a demonstration program in a short course, "Computer Methods of Optimum Structural Design" at the University of Missouri, Rolla, Missouri. It was also used in the Structural Design course at the University of Dayton and the Air Force Institute of Technology.

The purpose of this report is to generate comprehensive documentation for the "OPTSTAT" program.

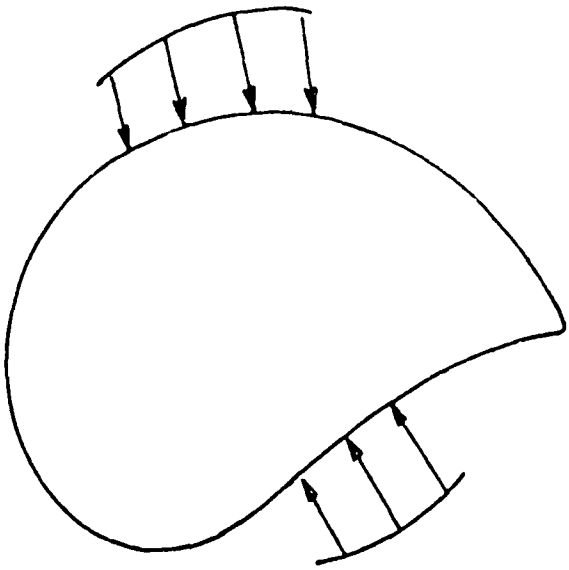
The program is based on the displacement method of finite element analysis[7-9]. In such an analysis the continuum is replaced by a discrete model consisting of a finite number of nodes connected by elements (See Figure 1). This discretization reduces the original differential equations of the continuum to a set of algebraic equations which can be solved much more readily on digital computers.

The program has basically four finite elements:

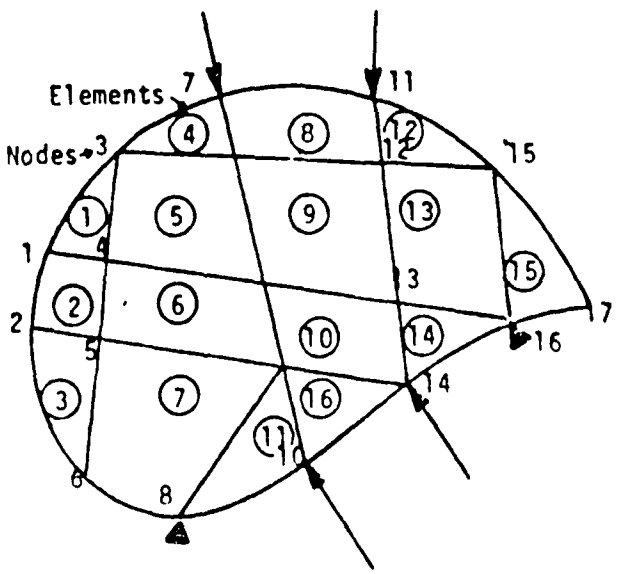
1. Bar (Axial Force Member)
2. Membrane Triangle
3. Membrane Quadrilateral
4. Shear Panel

The four elements and their local coordinate systems are shown in Figure 2. The bar is a constant strain line element and is equivalent to a rod element in the NASTRAN[10] program. The membrane triangle is a constant strain plate element similar to TRMEM in NASTRAN. The membrane quadrilateral is constructed out of four (non-overlapping) constant strain membrane triangles (element 2) with a fictitious interior node. This interior node is later removed by static condensation. This element is similar to QDMEM2 in NASTRAN. The shear panel is also constructed out of four non-overlapping triangles with a fictitious interior node. However, only the shear energy is considered in determining the stiffness of this element. Although the formulation is somewhat different, this element gives comparable results to the NASTRAN SHEAR element or the so called Garvey shear panel[11].

The basis for the derivation of the shear panel is emperical, and it is primarily intended to eliminate some of the difficulties encountered in using membrane triangles and quadrilaterals. For example, in beam problems (rectangular beams, I-beam, Box Beams including multicell wings and fuselage structures) the high stress gradients in the webs do not justify the use of constant strain triangles or quadrilaterals derived from these triangles. In fact, use of such elements for the webs (spars and ribs in wings) overestimates the stiffness by an order of magnitude.

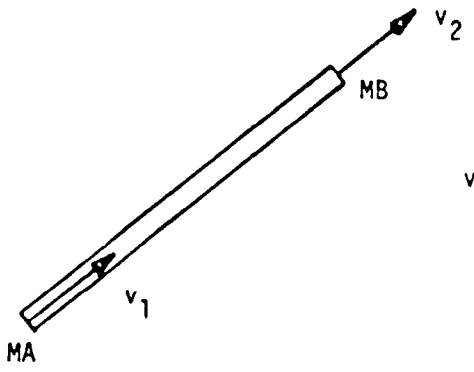


(a) Continuum

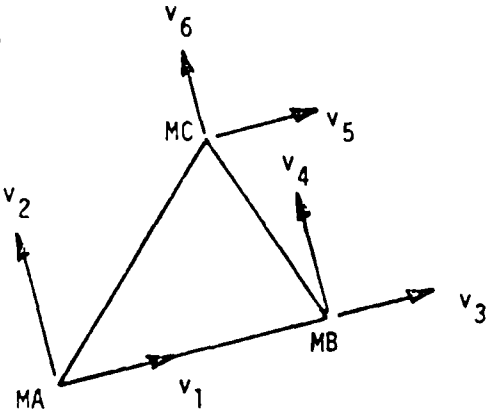


(b) Finite Element Model

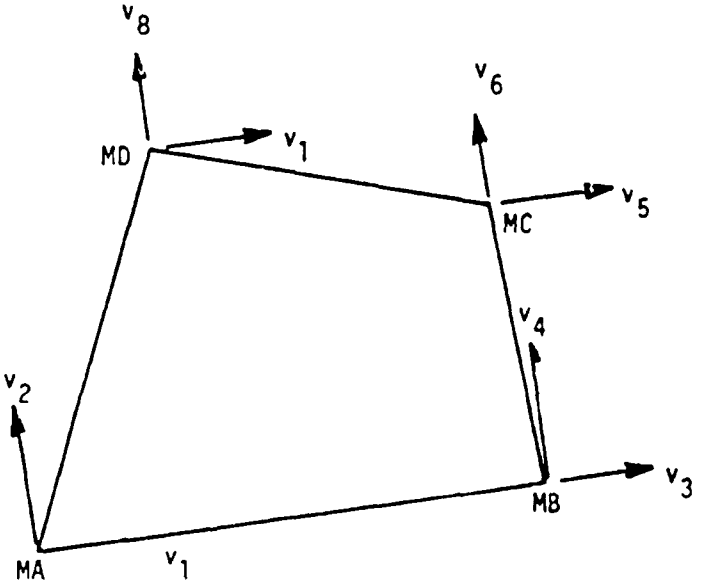
FIG. 1: Continuum and Finite Element Model



(a) Bar Element



(b) Triangular Membrane Element



(c) Quadrilateral or Shear Panel

FIG. 2: Elements and Local Coordinate System

Aerospace engineers have offset this difficulty to a large extent by judicious use of membrane elements in conjunction with the shear panels. In fact the early finite element models of wings and fuselages consisted primarily of bars and shear panels. However, the present practice of using membrane triangles and quadrilaterals for the top and bottom skins, bars for the posts, spar and rib caps, and shear panels for the spars and ribs eliminates to a large extent the need for determining the equivalent thicknesses and cross-sectional areas in the bars and shear panels model. The models consisting of these elements are most satisfactory for determining the primary load paths in built-up structures such as wings and fuselages. In addition the simplicity of these elements makes interpretation of the results easy and also keeps the analysis costs low because the stiffness matrices of these elements can be generated in a fraction of a second. The detailed formulation and additional information on these elements are given in Section 3.

In the finite element analysis a large proportion of the time is spent in the solution of the force displacement relations. The program uses standard Gaussian elimination with modifications to take into account the symmetry and sparsness characteristics of the stiffness matrix. The details of the solution scheme and storage of the stiffness matrix are given in Sections 2 and 6.

For optimization the program uses algorithms based on an optimality criteria[3]. Most of the optimality criteria algorithms are derived for stiffness type constraints. In particular "OPTSTAT" uses the optimality criteria derived for generalized stiffness and displacement constraints.

The generalized stiffness algorithm is used for stress constraint problems and the two together for stress and displacement constrained problems. The generalized stiffness algorithm is truly valid only when the stress limits are uniform for all the elements. However, it is being used as an approximation for variable stress limit problems. Variable stress limits arise because of different material, local buckling considerations etc. For isotropic constant stress elements the generalized stiffness algorithm reduces to the well known stress ratio algorithm.

"OPTSTAT" is an incore program whose core requirements depend on the problem size, primarily measured in terms of the number of degrees of freedom and the size of the semi-bandwidth. However, the bandwidth per se is not considered in the program. With an available core of about $120K_8$ one can solve problems of up to 200 to 300 degrees of freedom. With the full core of a machine like the CDC 6600, it is possible to solve problems of up to 900 degrees of freedom and a comparable number of elements. The details of core requirements are discussed in Appendix A.

The program is written in standard ANSI Fortran IV and is portable to most computers with this capability. It has been run on at least five different computers.

2. ANALYSIS

In the finite element analysis the continuum is replaced by a discrete model consisting of a finite number of nodes connected by elements (members). The rationale in such an approximation is that the response between the nodes (i.e. in the elements) can be expressed as a function of the response at the nodes. The functional relationship between the two responses is approximated by various interpolation functions or shape functions. The type of functions depends on the complexity of the problem at hand. This discretization reduces the original differential equations of the continuum to a set of algebraic equations which can be solved much more readily on digital computers.

The equations of the finite element analysis can be derived conveniently by considering the strain energy of the deformed system. For example, if the elastic body is idealized by m finite elements connecting q nodes (See Figure 1), the strain energy of the i^{th} element can be written as

$$\tau_i = \frac{1}{2} \int_{V_i} \underset{\sim}{\sigma}_i^{t*} \underset{\sim}{\varepsilon}_i \, dV \qquad (1)$$

where $\underset{\sim}{\sigma}_i$ and $\underset{\sim}{\varepsilon}_i$ are the stress and strain vectors and V_i is the volume of the element. For a linearly elastic body the relation between stress and strain can be written as

$$\underset{\sim}{\sigma}_i = \underset{\sim}{E}_i \, \underset{\sim}{\varepsilon}_i \qquad (2)$$

where $\underset{\sim}{E}_i$ is the symmetric matrix of material elastic constants. For typical plane stress problems the elastic constants matrix is of dimension 3x3. For

*Superscript t on a matrix represents transpose

an isotropic material in plane stress problems the elements of $\underset{\sim}{E}$ are as follows:

$$\underset{\sim}{E} = \frac{E}{1-\mu^2}\begin{bmatrix} 1 & \mu & 0 \\ \mu & 1 & 0 \\ 0 & 0 & \frac{1}{2}(1-\mu) \end{bmatrix} \tag{3}$$

where E and μ are the elastic modulus and poisson's ratio of the material respectively. For an orthotropic material the elastic constants matrix is given by

$$\underset{\sim}{E} = \frac{E_1}{1-\beta\mu^2}\begin{bmatrix} 1 & \mu\beta & 0 \\ \mu\beta & \beta & 0 \\ 0 & 0 & \frac{G}{E_1}(1-\beta\mu^2) \end{bmatrix} \tag{4}$$

where E_1 and E_2 are the longitudinal and transverse modulii respectively in the directions of the material property axes. β is the ratio of transverse to longitudinal modulus (E_2/E_1). G and μ are the shear modulus and poisson's ratio respectively.

The essence of the finite element approximation is that the internal displacements of the elements are expressed as functions of the displacements of the discrete nodes to which they are connected. The local coordinate systems and the nodal degrees of freedom of the four elements are shown in Figure 2. The functional relationship between the element internal displacements and the discrete nodal displacements is given by

$$\underset{\sim}{w}_i = \underset{\sim}{\phi}_i \, \underset{\sim}{v}_i \tag{5}$$

where the matrix $\underset{\sim}{w}_i$ represents the displacements in the element which are functions of the spatial coordinates (x, y). The shape function $\underset{\sim}{\phi}_i$ is a rectangular matrix, and its elements are also functions of the spatial coordinates. The vector $\underset{\sim}{v}_i$ represents the nodal displacements in the direction of the element degrees of freedom in the local coordinate system (Figure 2). Now the strain-displacement relations can be written as

$$\underset{\sim}{\varepsilon}_i = \underset{\sim}{B}\ \underset{\sim}{w}_i \tag{6}$$

where $\underset{\sim}{B}$ is a differential operator. For a plane stress problem $\underset{\sim}{B}$ is given by

$$\underset{\sim}{B} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \tag{7}$$

Substitution of Equations 2, 5 and 6 in 1 gives the expression for strain energy in the following form

$$\tau_i = \frac{1}{2}\ \underset{\sim}{v}_i^t\ \underset{\sim}{k}_i\ \underset{\sim}{v}_i \tag{8}$$

where $\underset{\sim}{k}_i$ is the element (member) stiffness matrix with respect to the discrete coordinates $\underset{\sim}{v}$ and is given by

$$\underset{\sim}{k}_i = \int_{V_i} \underset{\sim}{\phi}_i^t\ \underset{\sim}{B}^t\ \underset{\sim}{E}_i\ \underset{\sim}{B}\ \underset{\sim}{\phi}_i\ dV \tag{9}$$

An alternate but a convenient method of determining the elements of the member stiffness matrix is by invoking the principle of virtual work[12] which gives

$$1 \times k_{pq} = \int_{V_i} \underset{\sim}{\sigma}_i^{(p)^t} \underset{\sim}{\varepsilon}_i^{(q)} \, dV \tag{10}$$

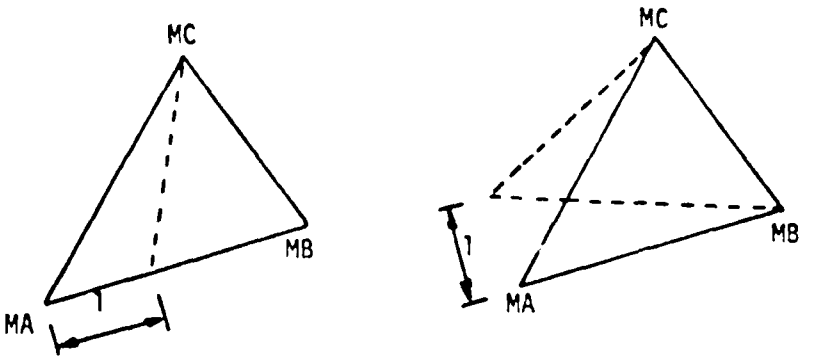
where $\underset{\sim}{\sigma}_i^{(p)}$ is the stress state due to the element displacement configuration in which $v_p = 1$ while all other v's are zero. Similarly $\underset{\sim}{\varepsilon}_i^{(q)}$ is the strain state due to the unit displacement configuration in the direction of the q^{th} degree of freedom. These two conditions are shown in Figure 3 for the degrees of freedom 1 and 2 of the membrane triangle. It should be noted that besides assuming appropriate shape functions, the integration in Equations 9 or 10 is one of the difficult tasks in the case of complex elements in finite element analysis. However, for membrane elements this integration does not present any difficulties as will be seen in the next section. For more complex elements the usual practice is to adopt numerical integration schemes[15,16].

From Equation 8 and Castigliano's first theorem, the relation between the element nodal forces and the displacements may be written as

$$\underset{\sim}{s}_i = \left[\frac{\partial \tau_i}{\partial v_j}\right] = \underset{\sim}{k}_i \, \underset{\sim}{v}_i \tag{11}$$

where $\underset{\sim}{s}_i$ is the element nodal force matrix corresponding to the displacement matrix $\underset{\sim}{v}_i$. Similar force-displacement relations for the total structure can be derived from the strain energy of the structure. The total strain energy Γ of the structure can be written as the sum of the energies of the individual components.

$$\Gamma = \sum_{i=1}^{m} \tau_i = \frac{1}{2} \sum_{i=1}^{m} \underset{\sim}{v}_i^t \, \underset{\sim}{k}_i \, \underset{\sim}{v}_i \tag{12}$$



(a) First Unit Mode (b) Second Unit Mode

FIG. 3: Examples of Unit Displacement Modes

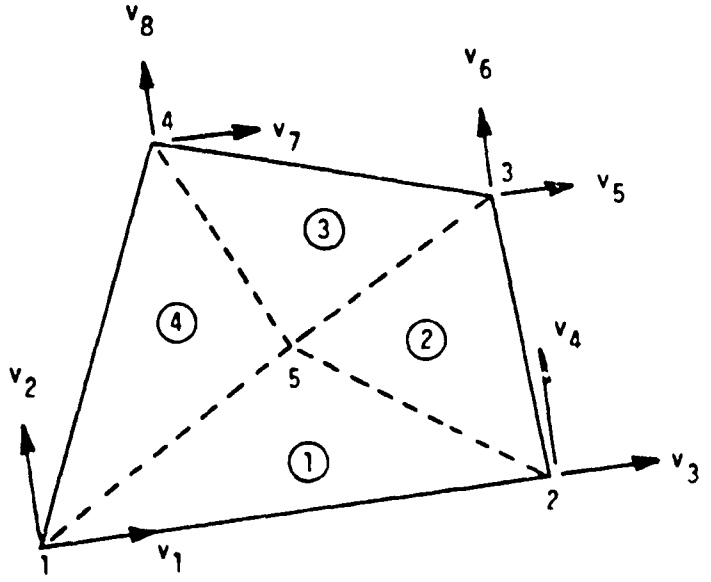


FIG. 4: Quadrilateral or Shear Panel Divided into Four Triangles

In general, for most structures, it is convenient to define a local coordinate system for each element and a global coordinate system for the total structure. In such a case the element and structure generalized coordinates can be related by

$$\underset{\sim}{v}_i = \underset{\sim}{a}_i \underset{\sim}{u} \tag{13}$$

where $\underset{\sim}{a}_i$ is the compatibility matrix. Its elements can be determined by kinematic reasoning alone provided the structure is kinematically determinate. The matrix $\underset{\sim}{u}$ is the generalized displacement vector of the structure in the global coordinate system. It is interesting to note that Equation 13 not only transforms element displacements from local to global coordinates but also gives information about how the elements are connected to the structure. From Equation 13 and the principle of virtual work it is easy to show that the transformation between the forces on the structure and the element internal forces is given by

$$\underset{\sim}{P} = \underset{\sim}{a}_i^t \underset{\sim}{s}_i \tag{14}$$

where $\underset{\sim}{P}$ is the force vector on the structure in the global coordinate system. The transformation given in Equation 14 is sometimes referred to as a contragradient transformation[15].

Substitution of Equation 13 in 12 gives the expression for the total strain energy in the form

$$\Gamma = \frac{1}{2} \underset{\sim}{u}^t \underset{\sim}{K} \underset{\sim}{u} \tag{15}$$

where $\underset{\sim}{K}$, the total stiffness matrix of the structure, is written as the sum of the component stiffness matrices.

$$\underset{\sim}{K} = \sum_{i=1}^{m} \underset{\sim}{a}_i^t \underset{\sim}{k}_i \underset{\sim}{a}_i \tag{16}$$

Again using Castigliano's first theorem the relation between the generalized force matrix $\underset{\sim}{P}$ corresponding to the displacement matrix $\underset{\sim}{u}$ may be written as

$$\underset{\sim}{P} = \left[\frac{\partial \Gamma}{\partial u_j}\right] = \underset{\sim}{K}\ \underset{\sim}{u} \tag{17}$$

In most structural analysis problems the stiffness matrix $\underset{\sim}{K}$ is sparsely populated. It is essential to take advantage of this fact in solving the load deflection equations (Equation 17), particularly in the case of problems with a large number of degrees of freedom where the cost of computation can be prohibitive otherwise. The "OPTSTAT" program uses Gaussian elimination with modifications to take into account the symmetry and sparseness of the stiffness matrix.

Basically Gaussian elimination involves decomposition of the stiffness matrix by

$$\underset{\sim}{K} = \underset{\sim}{L}\ \underset{\sim}{D}\ \underset{\sim}{L}^t \tag{18}$$

where $\underset{\sim}{L}$ is the unit lower triangular matrix and $\underset{\sim}{D}$ is a diagonal matrix. The advantage of this decomposition scheme is that the $\underset{\sim}{L}$ matrix retains some of the sparseness characteristics of $\underset{\sim}{K}$ which consequently reduces the number of computations. Also $\underset{\sim}{L}$ and $\underset{\sim}{D}$ can be assigned the same storage as $\underset{\sim}{K}$.

The next step is the forward substitution by

$$\underset{\sim}{L}\ \underset{\sim}{Y} = \underset{\sim}{P} \tag{19}$$

where the matrix $\underset{\sim}{Y}$ is given by

$$\underset{\sim}{Y} = \underset{\sim}{D}\ \underset{\sim}{L}^t\ \underset{\sim}{u} \tag{20}$$

In Equation 19 the solution of $\underset{\sim}{Y}$ can be accomplished by simple forward substitution. Once $\underset{\sim}{Y}$ is obtained, $\underset{\sim}{u}$ can be solved by back substitution using Equation 20. The last two steps together are generally referred to

as Forward-Back Substitution (FBS). Solution of Equation 17 for multiple load vectors involves the decomposition of the stiffness matrix once and repetition of FBS as many times as there are load vectors.

With the help of these basic equations the steps in the finite element analysis can be outlined as follows:

1. Input information consists of
    a. Geometry of the structure
        Node Coordinates
        Element Connections
        Section Properties
    b. Material properties
    c. Boundary conditions
    d. Loading
    e. Clues for appropriate (desired) output.
2. Element information consists of
    a. Determination of the local coordinate system for each element.
    b. Selection of the appropriate shape functions (Equation 5).
    c. Determination of the element stiffness matrix (Equation 9 or 10).
3. Transformation of the element stiffness matrix to the global coordinate system (Equation 16 without summation).
4. Determination of the structure stiffness matrix by summation of the component stiffnesses (Equation 16).
5. Incorporation of the boundary conditions.
6. Solution of the load-deflection equations (Equations 17, 18, 19 and 20).

7. Determination of the element displacements in their local coordinate system (Equation 13).

8. Determination of the stresses and energies in each element (Equations 6, 5, and 2).

The next section consists of the details of the stiffness matrix formulations for the four elements in this program.

3. FINITE ELEMENTS

The program "OPTSTAT" has four elements as mentioned earlier. They are all membrane elements. These four elements are generally adequate for determining the primary load paths of most aircraft structures. However, for a detailed stress analysis of local areas, higher order elements may be necessary.

BAR (ROD) ELEMENT

Basically this element is an axial force member. Its primary use is in two and three dimensional truss structures. It is also used extensively as spar and rib caps, posts around shear panels, stiffners and other line elements in aircraft structures. The local coordinate system of this element is shown in Figure 2. The positive x-axis is directed along the line joining the two ends. v_1 and v_2 represent the element end displacements. The corresponding two end forces are s_1 and s_2. The displacement field in the element is assumed to be linear which gives constant strain. For a linearly elastic material this assumption yields constant stress as well.

If w, the displacement at any point along the length of the bar, is given by

$$w = ax + b \tag{21}$$

where a and b are two undetermined coefficients and x is the coordinate of the point in the local coordinate system, then the end displacements v_1 and v_2 are given by

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \tag{22}$$

where x_1 and x_2 are the coordinates of the two ends in the local coordinate system. Then the shape function (Equation 5) corresponding to this linear displacement field can be written as

$$\underset{\sim}{\phi} = \frac{1}{(x_1 - x_2)} \left[(x - x_2), -(x - x_1)\right] \tag{23}$$

From the strain-displacement relations, the axial strain in the element is given by

$$\varepsilon_x = \frac{\partial w}{\partial x} = a \tag{24}$$

From the principle of virtual work (Equation 10) the individual elements of the member stiffness matrix can be written as

$$k_{ij} = \int_V \sigma_x^{(i)} \varepsilon_x^{(j)} \, dV = (-1)^{i+j} \frac{AE}{L} \tag{25}$$

where A is the cross-sectional area, L is the length of the member, and E is the modulus of elasticity of the material. The member stiffness matrix is given by

$$\underset{\sim}{k} = \frac{AE}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{26}$$

The member force matrix is given by

$$\underset{\sim}{s} = \underset{\sim}{k} \, \underset{\sim}{v} \tag{27}$$

The stress in the member is given by

$$\sigma_x = E \, \varepsilon_x \tag{28}$$

or

$$\sigma = \frac{s_1}{A} = \frac{-s_2}{A} \tag{29}$$

The strain energy in the element is given by

$$\tau_i = \frac{1}{2} \underline{s}^t \underline{v} \tag{30}$$

or

$$\tau_i = \frac{1}{2} \sigma_x \varepsilon_x A L \tag{31}$$

TRIANGULAR MEMBRANE ELEMENT

The membrane triangle is the basic plate element in the program. It is used to construct the membrane quadrilateral as well as the shear panel with some modifications. The membrane triangle can be used effectively in all cases where the primary loading is inplane forces. These include top and bottom skins of aircraft wings, flanges of I and box beams when they are subjected to constant normal stresses (tension or compression) only and skins of sandwich construction. However, they are not suitable for situations where high stress gradients exist. For example, they are unsuitable for spars and ribs of wings and other lifting surfaces, webs of I and box beams and flat plates where the primary load is bending. If used in such cases, they overestimate the stiffness or generate singularity. Figure 2 shows the triangle elements with the local coordinate system. The generalized coordinates v_1, v_2, ---, v_6 represent the inplane displacements of the three nodes in the local coordinate system. The displacement field in the element is assumed to be linear. This gives constant strain in the element. For a linearly elastic material the stress in the element will also be constant.

The linear displacement field in the element can be represented by

$$\begin{aligned} w_x &= a_1 x + b_1 y + c_1 \\ w_y &= a_2 x + b_2 y + c_2 \end{aligned} \tag{32}$$

where w_x and w_y are the x-y displacements in the plane of the plate in the local coordinate system. a_1, b_1 etc. are the six undetermined coefficients. Equation 32 can be written in matrix form as follows:

$$\underset{\sim}{w} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \end{bmatrix} \tag{33}$$

The six unknown coefficients can be uniquely determined by the six boundary conditions at the nodes.

$$\left[\begin{array}{c} v_1 \\ v_3 \\ v_5 \\ \hline v_2 \\ v_4 \\ v_6 \end{array}\right] = \left[\begin{array}{ccc:ccc} x_1 & y_1 & 1 & 0 & 0 & 0 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 0 & x_1 & y_1 & 1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \end{array}\right] \left[\begin{array}{c} a_1 \\ b_1 \\ c_1 \\ \hline a_2 \\ b_2 \\ c_2 \end{array}\right] \tag{34}$$

where x_1, y_1, ---, x_3 and y_3 are the coordinates of the three nodes of the triangle in the local coordinate system. It should be noted that the nodal displacements are grouped into x and y directions, so that the nodal coordinate matrix on the right hand side partitions into a diagonal matrix. The inversion of the partitioned diagonal matrix involves simply the inversion of the component matrix. Now the shape matrix $\underset{\sim}{\phi}$ is given by

$$\underset{\sim}{\phi} = \underset{\sim}{x}\ \underset{\sim}{Z}^{-1} \tag{35}$$

where the matrix $\underset{\sim}{x}$ is given by

$$\underset{\sim}{x} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix} \tag{36}$$

and the $\underset{\sim}{Z}$ matrix is given by

$$\underset{\sim}{Z} = \begin{bmatrix} \underset{\sim}{X} & 0 \\ 0 & \underset{\sim}{X} \end{bmatrix} \tag{37}$$

The coordinate matrix $\underset{\sim}{X}$ is given by

$$\underset{\sim}{X} = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} \tag{38}$$

It is interesting to note that each column of $\underset{\sim}{Z}^{-1}$ represents a unit displacement mode: i.e. the j^{th} column of the inverse represents a displacement mode in which $v_j = 1$ while all other nodal displacements are zero (See Figure 3). This fact is used to advantage in determining the elements of the member stiffness matrix.

From linear strain-displacement relations the strains can be written as

$$\varepsilon_x = \frac{\partial w_x}{\partial x} = a_1 \tag{39}$$

$$\varepsilon_y = \frac{\partial w_y}{\partial y} = b_2 \tag{40}$$

$$\varepsilon_{xy} = \frac{\partial w_x}{\partial y} + \frac{\partial w_y}{\partial x} = b_1 + a_2 \tag{41}$$

From the principle of virtual work (Equation 10) the elements of the member stiffness matrix can be written as

$$k_{ij} = \int_V \underset{\sim}{\sigma}^{(i)^t} \underset{\sim}{\varepsilon}^{(j)} \, dV = \int_V \underset{\sim}{\varepsilon}^{(i)^t} \underset{\sim}{E} \underset{\sim}{\varepsilon}^{(j)} \, dV \tag{42}$$

where $\underset{\sim}{\sigma}^{(i)}$ and $\underset{\sim}{\varepsilon}^{(j)}$ are the stress and strain matrices corresponding to the unit displacement modes explained under Equation 38. $\underset{\sim}{E}$ is the elastic constants matrix with respect to the element stiffness axis (See the local coordinate system of the triangular element in Fig. 2). If the material axis and the element stiffness axis coincide, $\underset{\sim}{E}$ would be the same as $\underset{\sim}{E}$ given in Equation 4 for orthotropic materials. In layered composite elements however, the material axis and the element local axis do not generally coincide and transformation of $\underset{\sim}{E}$ to the element local axis is necessary before using it in Equation 42. This transformation can be accomplished by considerations of energy invariance with axis rotation. For instance the element strain energy with respect to the material and the element local axes can be written as

$$\tau_m = \frac{1}{2} \underset{\sim}{\varepsilon}_m^t \underset{\sim}{E}_m \underset{\sim}{\varepsilon}_m \tag{43}$$

$$\tau_x = \frac{1}{2} \underset{\sim}{\varepsilon}^t \underset{\sim}{E} \underset{\sim}{\varepsilon} \tag{44}$$

where $\underset{\sim}{\varepsilon}_m$ is the strain matrix with reference to the material property axis. $\underset{\sim}{\varepsilon}$ is the strain matrix with reference to the element local axis. The strain matrices with reference to the material and element local axes are related by

$$\underset{\sim}{\varepsilon}_m = \underset{\sim}{T} \underset{\sim}{\varepsilon} \tag{45}$$

where $\underset{\sim}{T}$, the strain transformation matrix, is given by

$$\underset{\sim}{T} = \begin{bmatrix} \cos^2\theta & \sin^2\theta & \frac{1}{2}\sin 2\theta \\ \sin^2\theta & \cos^2\theta & -\frac{1}{2}\sin 2\theta \\ -\sin 2\theta & \sin 2\theta & \cos 2\theta \end{bmatrix} \tag{46}$$

and where θ is the angle between the element local axis and the material axis. By substituting Equation 45 in 43 and invoking the condition of energy invariance with axis rotation, the expression for the elastic constants transformation can be written as

$$\underset{\sim}{\varepsilon} = \underset{\sim}{T}^{t}\ \underset{\sim}{E}_{m}\ \underset{\sim}{T} \tag{47}$$

The linear displacement variation in Equation 32 implies constant strain, therefore the integral in Equation 4′ can be replaced by the volume of the element:

$$k_{ij} = \frac{1}{2}\ |\underset{\sim}{X}|\ t\ \underset{\sim}{\varepsilon}^{(i)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(j)} \tag{48}$$

where $|\underset{\sim}{X}|$ is the determinant of the nodal coordinate matrix which represents twice the area of the element and t is the thickness of the element. Now the stiffness matrix of the element is given by

$$\underset{\sim}{k} = \frac{1}{2}\ |\underset{\sim}{X}|\ t \begin{bmatrix} \underset{\sim}{\varepsilon}^{(1)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(1)} & \underset{\sim}{\varepsilon}^{(1)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(2)} \cdots\cdots & \underset{\sim}{\varepsilon}^{(1)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(5)} \\ \underset{\sim}{\varepsilon}^{(2)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(1)} & \underset{\sim}{\varepsilon}^{(2)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(2)} \cdots\cdots & \underset{\sim}{\varepsilon}^{(2)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(6)} \\ \vdots & \vdots & \vdots \\ \underset{\sim}{\varepsilon}^{(6)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(1)} & \underset{\sim}{\varepsilon}^{(6)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(2)} \cdots\cdots & \underset{\sim}{\varepsilon}^{(6)^{t}}\ \underset{\sim}{E}\ \underset{\sim}{\varepsilon}^{(6)} \end{bmatrix} \tag{49}$$

The stress matrix in the element is given by

$$\underset{\sim}{\sigma} = \underset{\sim}{E}\ \underset{\sim}{\varepsilon} \tag{50}$$

The stresses obtained by Equation 50 are with respect to the element local axis. It is often necessary to transform these to the material property axis. This transformation can be obtained by

$$\underset{\sim}{\sigma}_{m} = \underset{\sim}{T}_{s}\ \underset{\sim}{\sigma} \tag{51}$$

where $\underset{\sim}{\sigma}_m$ is the stress matrix with respect to the material axis. The stress transformation matrix from the element local axes to the material axis is given by

$$\underset{\sim}{T}_s = \begin{bmatrix} \cos^2\theta & \sin^2\theta & \sin 2\theta \\ \sin^2\theta & \cos^2\theta & -\sin 2\theta \\ -\frac{1}{2}\sin 2\theta & \frac{1}{2}\sin 2\theta & \cos 2\theta \end{bmatrix} \tag{52}$$

The member force matrix is given by

$$\underset{\sim}{s} = \underset{\sim}{k}\ \underset{\sim}{v} \tag{53}$$

The strain energy in the element is given by

$$\tau_i = \frac{1}{4}\,|\underset{\sim}{X}|\ t\ \underset{\sim}{\sigma}^t\ \underset{\sim}{\varepsilon} \tag{54}$$

or

$$\tau_i = \frac{1}{2}\,\underset{\sim}{s}^t\ \underset{\sim}{v} \tag{55}$$

The next important step in the evaluation of the stress state in an element is the selection of a suitable failure criteria because of the combined stresses (σ_x, σ_y and σ_{xy}) in plate elements. For isotropic materials the energy of distortion or the Von-Mises criterion is accepted as most satisfactory. The effective stress according to this criterion is given by

$$\sigma_{eff} = (\sigma_x^2 + \sigma_y^2 - \sigma_x\,\sigma_y + 3\sigma_{xy}^2)^{1/2} \tag{56}$$

When the allowable stresses are different in different directions, the effective stress ratio (ESR) according to the modified energy of distortion criterion can be obtained by

$$ESR = \left[\left(\frac{\sigma_x}{XX}\right)^2 + \left(\frac{\sigma_y}{YY}\right)^2 - \left(\frac{\sigma_x\sigma_y}{XXYY}\right) + \left(\frac{\sigma_{xy}}{ZZ}\right)^2\right]^{1/2} \tag{57}$$

where XX and YY are the tension or compression allowable in the x and y directions respectively, and ZZ is the shear allowable. Then the margin of safety (MS) is determined by

$$MS = \frac{1-ESR}{ESR} \tag{58}$$

The requirement of a positive margin of safety constitutes a stress constraint in optimization.

The failure criterion as given by Equation 57 is adequate for isotropic as well as equivalent orthotropic structures. However, in the case of fiber reinforced layered composite materials, the question becomes much more complicated and there is little agreement on the type of criterion to be used. The fiber failure, matrix failure, delamination, and the effects of cut outs and bolt holes can trigger different failure modes. It is difficult, if not impossible, to combine all these effects into a single neat failure criterion as in metal structures. The present practice consists of a number of emperical criteria whose justification sometimes appears to be more emotional than rational. A review of some of these criteria is given in References[16,17]. The "OPTSTAT" program uses the failure criterion given by Equation 57 for isotropic and equivalent orthotropic structures. For layered composite structures the fiber failure is used as a failure criterion. However, it is a relatively simple matter to modify this criterion to suit other requirements.

The composite element in "OPTSTAT" consists of stacked orthotropic membrane elements. Each orthotropic element (layer) in the stack represents the combined effect of all the fibers in one direction. The stiffness of the composite element is obtained by adding the stiffnesses of the component

orthotropic elements representing all the fiber directions. This addition of the stiffnesses can be written as

$$\underset{\sim}{k} = \sum_{j=1}^{\ell} \underset{\sim}{k}_j \tag{59}$$

where $\underset{\sim}{k}_j$ represents the stiffness of all the fibers in one direction and ℓ represents the number of fiber directions in the composite element. The matrix $\underset{\sim}{k}_j$ for each direction of fibers is determined by Equation 49. It is also assumed, for the summation in Equation 59 to be valid, that the stiffness matrices $\underset{\sim}{k}_j$ in each composite element are determined with respect to the same set of reference axis such as the local element axis.

The composite element in "OPTSTAT" has at present a provision for four fiber orientations. These fiber orientations are 0°, 90°, and $\pm$45°. It is further assumed that the composite element is made of a balanced laminate. By adjusting the relative percentages of the fibers, the optimum directional properties of the laminate can be obtained. In assessing the failure of the laminate a weighted average of the effective stress ratios is considered instead of the failure of the individual fibers. This weighted average ESR is computed by

$$ESR = \alpha_0\, ESR_0 + \alpha_{90}\, ESR_{90} + \alpha_{45}\, ESR_{45} + \alpha_{-45}\, ESR_{-45} \tag{60}$$

where α_0, α_{90}, α_{45} and α_{-45} are the percentage of fibers in the 0°, 90° and $\pm$45° respectively. Similarly ESR_0, ESR_{90}, ESR_{45} and ESR_{-45} are the effective stress ratios of the 0°, 90°, 45° and -45° layers.

QUADRILATERAL MEMBRANE ELEMENT

The quadrilateral element is most frequently used to represent membrane skins unless the corners etc. require the use of the triangular element. Figure 4 shows the local coordinate system and the generalized coordinates

(displacements) v_1 through v_8. The element is assumed to be a flat plate, and all nodes are assumed to lie on a plane connecting the first three nodes (1, 2, and 3). In effect the warping in the element is ignored. This approximation results in an overestimation of the stiffness of a truly warped quadrilateral element. In most cases the effect of the approximation is small, and it can be further reduced by reducing the mesh size of the model in the regions of high warping. However, if the warp is too large, the quadrilateral should be broken up into two or more triangles.

As mentioned earlier, the stiffness of the quadrilateral element is determined by breaking it into four component triangles as shown in Figure 4. A fictitious node in the quadrilateral is located by averaging the coordinates of the four nodes as given by

$$x_5 = \frac{x_1 + x_2 + x_3 + x_4}{4} \tag{61}$$

$$y_5 = \frac{y_1 + y_2 + y_3 + y_4}{4} \tag{62}$$

The stiffness of the four triangles is then computed by Equation 49 in the local coordinate system shown in Figure 2c. Addition of the four stiffness matrices gives a 10 x 10 stiffness matrix with two degrees of freedom included for the fifth node. This fictitious node is later removed by static condensation before adding to the total structure. The procedure for static condensation is outlined next.

The force displacement relations of the 5 node quadrilateral are written as

$$\underset{\sim}{R}_Q = \underset{\sim}{k}_Q \underset{\sim}{r}_Q \tag{63}$$

where the subscript refers to the quadrilateral element with 5 nodes. Equation 63, partitioned to isolate the degrees of freedom of the fifth node, can be written as

$$\begin{bmatrix} \underset{\sim}{R}_I \\ \underset{\sim}{R}_{II} \end{bmatrix} = \left[\begin{array}{c|c} \underset{\sim}{K}_{I,\,I} & \underset{\sim}{K}_{I,\,II} \\ \hline \underset{\sim}{K}_{II,\,I} & \underset{\sim}{K}_{II,\,II} \end{array}\right] \begin{bmatrix} \underset{\sim}{r}_I \\ \underset{\sim}{r}_{II} \end{bmatrix} \tag{64}$$

Equation 64 can be written as two separate equations

$$\underset{\sim}{R}_I = \underset{\sim}{k}_{I,I}\ \underset{\sim}{r}_I + \underset{\sim}{k}_{I,II}\ \underset{\sim}{r}_{II} \tag{65}$$

$$\underset{\sim}{R}_{II} = \underset{\sim}{k}_{II,I}\ \underset{\sim}{r}_I + \underset{\sim}{k}_{II,II}\ \underset{\sim}{r}_{II} \tag{66}$$

Since the fifth node does not actually exist in the original model, no external forces can be applied to this node. This condition gives

$$\underset{\sim}{r}_{II} = -\underset{\sim}{k}^{-1}_{II,II}\ \underset{\sim}{k}_{II,I}\ \underset{\sim}{r}_I \tag{67}$$

Substitution of Equation 67 in 65 gives

$$\underset{\sim}{R}_I = (\underset{\sim}{k}_{I,I} - \underset{\sim}{k}_{I,II}\ \underset{\sim}{k}^{-1}_{II,II}\ \underset{\sim}{k}_{II,I})\ \underset{\sim}{r}_I \tag{68}$$

From Equation 68 the stiffness matrix of the original quadrilateral can be written as

$$\underset{\sim}{k} = \underset{\sim}{k}_{II} - \underset{\sim}{k}_{I,II}\ \underset{\sim}{k}^{-1}_{II,II}\ \underset{\sim}{k}_{II,I} \tag{69}$$

The stiffness as obtained by Equation 69 is added to the total structure after appropriate coordinate transformations to the global coordinate system. When the structure displacements are determined, the fifth node displacements can be determined by Equation 67. Now the stresses in each triangle can be determined as before. The effective stress ratio is determined for each triangle separately (Equation 57), and then a weighted average is used in computing the effective stress ratio and the margin of safety. This weighted

average is computed by

$$ESR = \frac{(ESR)_1 \Delta_1 + (ESR)_2 \Delta_2 + (ESR)_3 \Delta_3 + (ESR)_4 \Delta_4}{\Delta_1 + \Delta_2 + \Delta_3 + \Delta_4} \qquad (70)$$

where $(ESR)_1$ thru $(ESR)_4$ are the effective stress ratios of the four triangles. Δ_1 thru Δ_4 are the respective planform areas of the triangles. In the case of fiber reinforced composite elements a further averaging across the thickness of the elements is used, as in Equation 59, in determining the effective stress ratio. Now the margin of safety MS is computed as before by Eq. 58.

SHEAR PANEL

As the name indicates the shear panel is devised for the purpose of representing shear transmitting elements. For example in wing structures the top and bottom skins can be represented by membrane (triangle and quadrilateral) elements. If the same elements are used for spars and ribs, the resulting finite element model grossly overestimates the stiffness of the structure. What this means is that the displacements obtained by this model will be much smaller, or if this model is used for dynamic analysis, the frequencies of the structure will be much higher and cannot be matched with the results obtained from ground vibration tests. This behavior is due to the assumption of constant strain (stress) in the membrane element formulations. Most web elements in box or I-beams carry primarily shear and some normal stresses. In other words their deformation is primarily due to shear and not due to normal stresses. The normal stresses in webs usually have steep stress gradients, and the assumption of constant stress (or strain) is not justified. To offset this difficulty, and yet preserve the simplicity of the constant strain elements, a shear panel was formulated (Reference) with the assumption that it carries only shear stresses. The bars and other

membrane elements that surround the shear panel are supposed to carry the normal stresses. Such a situation does not actually exist in reality, and thus the shear panel is an emperical element. However, the models built on such an assumption appear to produce satisfactory results.

Until recently it was a common practice in aircraft companies to model wings, fuselages, and empennage structures simply by bars and shear panels to obtain primary load path information. In such idealizations it was a common practice to assign a third of the cross-sectional area as spar and rib caps and the remainder for the shear panels. It should be pointed out that every shear panel must be surrounded on all four sides by normal stress carrying elements such as bars or membrane or bending elements. If the natural model does not contain such an element on any side of the shear panel, a nominal (or fictitious) bar (post) must be provided. Otherwise the model will have a singularity.

The shear panel in "OPTSTAT" is constructed out of four triangles with the fictitious node inside as in the membrane quadrilateral discussed earlier. However, the stiffness matrices of the component triangles are determined by considering only the shear strain energy (Equation 48).

$$k_{ij} = \frac{1}{2} |\underset{\sim}{X}| \; t \; \epsilon_{xy}^{(i)} \; G \; \epsilon_{xy}^{(j)} \tag{71}$$

where G is the shear modulus, and $\epsilon_{xy}^{(i)}$ and $\epsilon_{xy}^{(j)}$ are the shear strains due to the unit displacement modes discussed earlier. There is one point that must be made here. The shear stress (strain) in an element changes with the orientation of the reference axis. Thus the stiffness matrix of the element can be sensitive to the reference axis. For rectangular elements the shear strain energy would be the same regardless of which side is selected for the

reference axis. However, for quadrilaterals the stiffness matrix does depend on the reference axis. The errors produced by such departures are usually not significant, but it is worthwhile to make note of the assumptions involved. The OPTSTAT program has a provision for specifying any one of the four sides of the quadrilateral as the reference axis.

As in the quadrilateral element the shear stresses in all four triangles are determined separately but with respect to the same reference axis. Of course, the normal stresses in the shear panels have no meaning. The margin of safety is determined by a weighted average of the effective stress ratios (ESR) as in the quadrilateral. The strain energy is determined by considering only the shear stress and strain. It should be noted that the shear panel can be used only as an isotropic or equivalent isotropic element.

4. OPTIMALITY CONDITIONS

4.1 General

The optimization method is explained here in the context of the procedure used in the computer program "OPTSTAT". Basically this procedure consists of two steps. The first involves derivation of optimality conditions and associating them with an energy condition in the structure. In the second step, an iterative algorithm is derived with the help of the energy condition to achieve the optimality.

The weight of the structure is the objective function to be minimized and it is given by

$$W(A) = \sum_{i=1}^{m} \rho_i A_i \ell_i \tag{72}$$

where ρ_i is the weight density and the product $A_i \ell_i$ represents the volume of the element. The vector $\underset{\sim}{A}$ represents the design variables and they are the only quantities that change in the optimization. The constraint conditions are given by

$$G_i (\underset{\sim}{A}) = G_i (A_1, A_2 \ldots .A_m) \leq G_{io} \quad i = 1,2,\ldots.p. \tag{73}$$

and

$$\underset{\sim}{A}^{(L)} \leq \underset{\sim}{A} < \underset{\sim}{A}^{(u)} \tag{74}$$

The first set is considered as response (behavioral) constraints and the second as size constraints. The vectors $\underset{\sim}{A}^{(L)}$ and $\underset{\sim}{A}^{(u)}$ are the lower and upper limits respectively on the sizes.

The Lagrangian formulation for constrained minimization can be written as

$$\phi(\underset{\sim}{A}) = W(\underset{\sim}{A}) + \sum_{i=1}^{p} \lambda_i \psi_i (\underset{\sim}{A}) \tag{75}$$

where $\phi(\underset{\sim}{A})$ is the Lagrangian function and λ's are the Lagrangian multipliers. It is assumed that there are p constraints and they are represented by

$$\psi_i(\underset{\sim}{A}) = G_i(\underset{\sim}{A}) - G_{io} \leq 0 \tag{76}$$

It should be noted that the constraint set p includes only the response constraints and not the size constraints.

Minimization of the Lagrangian ϕ with respect to the design variable vector $\underset{\sim}{A}$ gives the condition for the stationary value of the objective function with the constraint conditions ψ as

$$\frac{\partial \phi}{\partial A_i} = \frac{\partial}{\partial A_i}[W(\underset{\sim}{A})] + \sum_{j=1}^{p} \lambda_j \frac{\partial}{\partial A_i}[\psi_j(\underset{\sim}{A})] = 0 \tag{77}$$

From equation 77 the optimality condition can be written as follows:

$$\sum_{j=1}^{p} e_{ij} \lambda_j = 1 \quad i = 1, 2, \ldots\ldots m \tag{78}$$

The m such equations corresponding to the m design variables can be written in the matrix form as follows:

$$\underset{\sim}{e}\ \underset{\sim}{\lambda} = \underset{\sim}{1} \tag{79}$$

The elements of matrix $\underset{\sim}{e}$ are given by

$$e_{ij} = \frac{\frac{\partial}{\partial A_i}[\psi_j(\underset{\sim}{A})]}{\frac{\partial}{\partial A_i}[W(\underset{\sim}{A})]} \tag{80}$$

Equation 80 represents the ratio of constraint to objective functions gradients with respect to the design variables. These ratios can be associated with special forms of energy densities depending on the type of constraint functions. This aspect will be discussed later in

connection with three types of constraints.

The solution of the optimization problem involves (m+p) unknown quantities, where m is the number of design variables and p is the number of Langrangian multipliers corresponding to p constraint conditions. However, Eq. 79 represents only m equations. The additional p equations can be obtained by writing the original constraint conditions as follows:

$$\sum_{i=1}^{m} e_{ij}\, \rho_i\, v_i = G_{jo} \qquad j = 1, 2, \ldots, p \tag{81}$$

Combining Eqs. 79 and 81 gives the necessary equations for determining the Lagrangian multipliers as follows:

$$\underset{\sim}{H}\, \underset{\sim}{\lambda} = \underset{\sim}{G}_o \tag{82}$$

where the matrix $\underset{\sim}{H}$ is given by

$$\underset{\sim}{H} = \underset{\sim}{e}^t\, \bar{\underset{\sim}{A}}\, \underset{\sim}{e} \tag{83}$$

$\bar{\underset{\sim}{A}}$ is a diagonal matrix and its i^{th} diagonal element is given by

$$\bar{A}_{ii} = \rho_i v_i \tag{84}$$

The elements of matrix $\underset{\sim}{H}$ cannot be determined explicitly because the $\underset{\sim}{e}$ and $\bar{\underset{\sim}{A}}$ matrices are functions of the design variable vector $\underset{\sim}{A}$ which is itself unknown. Eqs. 79 and 82 are nonlinear sets of equations, and they can be solved only by iterative methods.

To reduce some of the difficulties involved in solving the nonlinear sets of equations a number of simplifying assumptions were made in constructing iterative algorithms in the program "OPTSTAT". For

instance specialization of Eq. 82 to a single constraint and some simplifying assumptions give a simple expression for λ as follows for most stiffness type constraints.

$$\lambda = \frac{W}{G_o} \tag{85}$$

If the λ's are used simply as weighting parameters, then in multiple constraints their value can be approximated by

$$\lambda_i = \frac{W}{G_{io}} \tag{86}$$

This simplification eliminates the need for the solution of Eq. 16 and the associated positive negative λ's dilemma. Using this approach a number of truss structures were optimized[3]. There are a number of other ways indicated in the literature.

4.2 Specialization to Generalized Stiffness.

If $\underset{\sim}{R}$ and $\underset{\sim}{r}$ are the generalized force (external) and the corresponding displacement vectors, the generalized stiffness constraint will be defined as

$$G_i(\underset{\sim}{A}) = \frac{1}{2}\ \underset{\sim}{R}^t_i\ \underset{\sim}{r}_i \qquad i = 1, 2, \ldots\ldots p \tag{87}$$

The p constraints correspond to p independent loading conditions $\underset{\sim}{R}_i$.

Substitution of equation 87 in 80 gives the expression for e_{ij} as

$$e_{ij} = \frac{1}{2}\ \frac{\underset{\sim}{r}^t_j\ \underset{\sim}{K}_i\ \underset{\sim}{r}_j}{\rho_i\ A_i\ \ell_i} \tag{88}$$

Where e_{ij} is the strain energy density in the element i due to the loading condition j.

The optimality condition for the generalized stiffness involving multiple loading conditions can be stated from equation 79 as follows:

"The weighted sum of the strain energy densities corresponding to multiple loading conditions should be equal to unity in all the elements." The Lagrangian multipliers are the weighting parameters.

4.3 Specialization to Displacement Constraints

In the case of displacement constraints in the direction of specified degrees of freedom the constraint condition will be defined as

$$G_j(\underset{\sim}{A}) = \underset{\sim}{F}^t_j \, \underset{\sim}{r} \qquad j = 1, 2, \ldots . p \tag{89}$$

where $\underset{\sim}{F}_j$ is the virtual load vector corresponding to the j^{th} displacement constraint. The elements of the vector $\underset{\sim}{F}_j$ are all zero except in the direction of the constrained degree of freedom. The value in that direction is unity. The vector $\underset{\sim}{r}$ is the displacement vector due to the applied load. The displacements due to virtual load vectors will be designated by $\underset{\sim}{f}$. Now substitution of equation 89 in 88 gives the expression for e_{ij} as

$$e_{ij} = \frac{\underset{\sim}{f}^t_j \, \underset{\sim}{K}_i \, \underset{\sim}{r}}{\rho_i \, A_i \, \ell_i} \tag{90}$$

where e_{ij} is the virtual strain energy density in the i^{th} element corresponding to the j^{th} constraint condition. The optimality condition for multiple displacement constraints can be stated as follows:

"The weighted sum of the virtual strain energy densities corresponding to multiple displacement constraints should be equal to unity in all the elements". The Lagrangian multipliers are once again the weighting parameters.

From the optimality conditions derived in this section, the element resizing algorithms can be derived directly. The form of the resizing algorithm is as follows:

$$\underset{\sim}{A}^{\nu+1} = \underset{\sim}{A}^{\nu} \left[\sum_{j=1}^{b} C_j \, e_{ij}^{\nu} \right]^{1/2} \tag{91}$$

where c_j are the weighting parameters which can be approximated as functions of Lagrangian multipliers. The sizes of the elements as well as the percentage fiber orientations in each element are determined by such an energy based algorithm. The details of the implementation of the algorithm are given in Reference 2.

5. ORGANIZATION OF THE PROGRAM

The material presented in this section is intended either to help introduce changes into the program or to expand its scope for the specific needs of a researcher as the authors have done in the past ten years. The steps outlined at the end of Section 2 are summarized in the flow-chart in Figure 5. There are a total of 15 boxes in the flow-chart. Each of these boxes generally involves one or more subroutines. The subroutines that belong to each of these boxes are identified first, then the function of each subroutine will be discussed in the next section with the help of the equations given in Sections 2, 3, and 4.

Box 1 - Input

Input in the present version of the "OPTSTAT" program is not in subroutine form. However, the input statements are all at the beginning of the program, and thus they can be grouped into a single subroutine.

For example, it is relatively easy to write a subroutine with NASTRAN type input. The description of the various arrays (See input instructions) and their dimension requirements given in Appendix A can be quite helpful in writing such an input routine.

Box 2 - Map Stiffness Matrix

This step involves a single subroutine called "POP". The purpose of this routine is simply to estimate the storage requirements of the stiffness matrix and to map its profile. The stiffness matrix is stored in a single array called SK. The elements of the matrix are stored columnwise

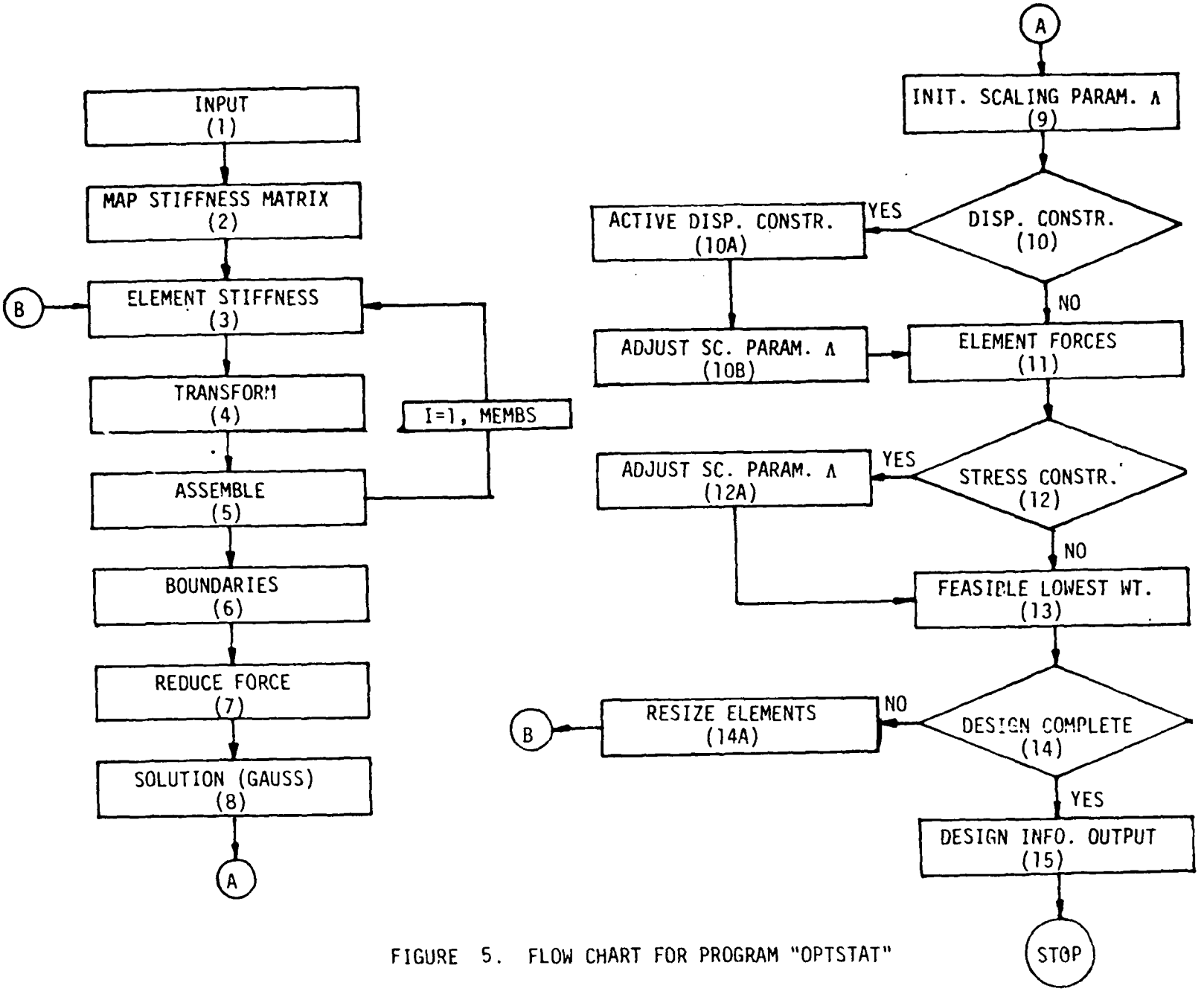


FIGURE 5. FLOW CHART FOR PROGRAM "OPTSTAT"

starting from the first non-zero element in the column to the diagonal element. Since the matrix is symmetric, only the upper triangle is stored.

Box 3 - Element Stiffness

There are four elements in the program. All of them require the subroutines "COORD" and "PREPAR". The subroutine "TRECON" is required if the material is orthotropic. The orthotropic material can be used only triangular and quadrilateral elements. In addition all the plate elements require the routine "ELSTIC". The remaining subroutines are listed separately for each element.

i. Bar (Rod) Element:

The bar element is shown in Figure 2a with the local coordinate system and degrees of freedom. This element requires the subroutine "ELSTIF" which generates the bar stiffness matrix in the local coordinate system and also transforms it to the global coordinate system.

ii. Triangular Membrane Element:

The element and its local coordinate system are shown in Figure 2b. The subroutine "PLSTIF" is the only other routine required by this element. It generates the stiffness matrix of the triangle in the local coordinate system.

iii. Quadrilateral Membrane Element and Shear Panel

The elements and their local coordinate system are shown in Figure 2c. The subroutines "QDRLTL", "PLSTIF", "SUM", "CONDNS", "CHANGE" and "CRAMER" are the additional routines required by these elements. Together these subroutines generate the stiffness matrix of either the quadrilateral membrane or shear panel. The routine "QDRLTL" calls "PLSTIF", "SUM" and "CONDNS". The routine "PLSTIF" calls "CRAMER". Similarly "CONDNS" calls "CHANGE".

Box 4 - Transform

This step involves a single subroutine called "TRNSFM". It transforms the stiffness matrices of the triangles, quadrilaterals, and shear panels from the local to the global coordinate system.

Box 5 - Assemble

"ASEMBL" is the only subroutine used in this step. Its purpose is to add the element stiffness matrices to the total stiffness matrix of the structure. The steps 3 thru 5 form a loop in which all the element stiffness matrices are computed and assembled into the total stiffness matrix.

Box 6 - Boundaries

The routine called "BOUND2" eliminates the rows and columns of the stiffness matrix corresponding to the support degrees of freedom of the structure. In addition it also condenses the stiffness matrix.

Box 7 - Reduce Force

This step involves a routine called "REDUCE". It eliminates the rows of the force matrix corresponding to the support degrees of freedom.

Box 8 - Solution of the Force Deflection Equations

The routine "GAUSS" solves the load deflection equations by Gaussian elimination. A large percentage of the analysis time (80 to 90%) is spent in this routine, and its efficiency is extremely important in reducing the costs of the analysis. At the end of this step the displacements of the structure are available in condensed form (excluding boundary degrees of freedom) in the global coordinate system.

Box 9 - Initial Scaling Parameter Λ

The analysis in the "OPTSTAT" program is made with the relative design vector, which is obtained by normalizing the design vector with the largest value of the design variable. The actual design variables are determined by scaling through to the constraint surface. The scaling is done by adjusting the parameter Λ. The initial value of this scaling parameter is obtained by equating the total strain energy to the energy capacity of the structure[1]. The energy capacity is defined (arbitrary) as the product of the material volume of the structure times the square of the allowable strain. The scaling parameter represents the product of the largest value of the design variable times the modules of elasticity of a reference material. One of the materials of the structure is arbitrary assigned as the reference material.

Box 10 - Displacement Constraints

If the design conditions specify displacement constraints, the active constraints are determined in step 10A and the scaling parameter Λ is adjusted in step 10B. The purpose of this step is to scale the design to satisfy displacement constraints. All the operations in this step are included in the main program and no subroutines are involved.

Box 11 - Element Forces

The program determines the stresses in all the elements in this step. In addition to strain energies in the elements are also determined in this step. The element forces are not actually determined. The stresses are determined directly from element displacements. The subroutines used in this step depend on the type of element involved. The subroutines "COORD", "PREPAR" and "ELFORC" are required by all the elements. In addition all the plate elements require the routine "ELSTIC". The

subroutine "TRECON" is required if the material is orthotropic. The orthotropic material can be used only for triangular and quadrilateral elements. The remaining subroutines are listed separately for each element.

i. Bar (Rod) Element:

The stress in this element is computed in the program itself. No additional routines are involved. At the same time the element strain energy is also computed.

ii. Triangular Membrane Element:

The subroutines "STRESS" and "CRAMER" are involved in this step. The routine "STRESS" calls "CRAMER". The purpose of this routine is to calculate stresses in the triangular element. In addition this routine calculates strain energy and the effective stress in the element (See Equations 44 and 45).

iii. Quadrilateral Membrane and Shear Panel

This step involves routines "QDRLTL", "PLSTIF", "SUM", "CONDNS", "CRAMER", "QLSTRS" and "STRESS". It should be noted that the routine "QDRLTL" calls "PLSTIF", "SUM" and "CONDNS". "PLSTIF" in turn calls "CRAMER".

Box 12 - Stress Constraints

It is assumed in the "OPTSTAT" program that constraints are part of all structural design problems. If they are active the scaling parameter adjusted in Box 12A. At the end of step 12 a completely feasible design is available. No subroutines are involved in this step.

Box 13 - Feasible Lowest Weight

The weight of the structure is determined in this step. Also the weights of the four groups of elements are determined at the same time.

Box 14 - Design Complete

The decision whether the design is complete is based on the number of cycles of iteration specified in stress and displacement constraint modes. "OPTSTAT" designs the structures in two modes. First it resizes the elements in stress constraint mode. This resizing is continued as long as there is reduction in weight. There it enters displacement constraint mode if there are constraints on displacements.

The resizing in stress constraint mode is relatively simple because it is based on strain energy density of the elements. The strain energy in the elements is already determined in step 11. The only subroutine used in this case is "LMSIZE" in case of composite elements. This routine determines percentage fibers in each direction.

Resizing in displacement constraint mode is more involved. Resizing in this step is based on the virtual strain energy of the elements. To determine the virtual energy in the elements the structure has to be analyzed with the virtual loads. The latter steps involves repetition of forward-ba substition of the Gausian elimination. The subroutines "REDUCE", "GAUSS1" and "RESTOR" are used for determining the virtual displacements of the structure. For determining virtual strain energies the routines "COORD", "ELFORC", and "PREPAR" are required for all the elements. In addition all the plate elements require the routines "ELSTIC" and "UNITEG". The subroutine "TRECON" is required in case of orthotropic materials. The subroutine "LMSIZE" is required for all layered composite elements. The remaininq subroutines are listed separately for each element.

i. Bar (Rod) Element:

The virtual strain energy in this element is determined in the program itself. No additional subroutines are involved.

ii. Triangular Membrane Element:

The subroutines "PLSTIF" and "CRAMER" are the additional routines required. "PLSTIF" routine calls "CRAMER".

iii. Quadrilateral Membrane and Shear Panel.

The subroutines "ADRLTL", "PLSTIF", "SUM", "CONDNS" and "CHANGE" are the additional routines required by these elements. The routine "QDRLTL" calls "PLSTIF", "SUM" and "CONDNS". The routine "PLSTIF" calls "CRAMER". Similarly "CONDNS" calls "CHANGE".

Box 15 - Output - Design Information

The output of the program consists of element information and the nodal information. The subroutines used for computing the element information are the source as in step 11. For nodal information the subroutine "PRNTDR" is used. The section on "OUTPUT" gives the details of the format of the output.

The two subroutines "LACALC" and "LAYPR" convert the percentage fibers in each direction to the nearest discrete number of layers in case of composite elements.

In addition to the above 15 steps there are instructions for weight computations and other details, and their purpose can be identified from the program. There are very few comment cards in the main body of the program and this omission is by design in order to avoid continuous updating. The user can incorporate his own comment cards with the help of the explanation given in this section.

6. DESCRIPTION OF THE SUBROUTINES

"ANALYZE" consists of the main program and 21 Subroutines. The main program has 260 cards. The length of the Subroutines varies from 15 to 62 cards. The total length of the program is under 1000 cards. A list of the Subroutines, the number of Cards in each Subroutine and other details are given in Table 1. The flow chart, Fig. 5, and the explanation in the previous section give details of the main program. The description of the Subroutines is given in the remainder of this section.

Subroutine "POP"

The purpose of Subroutine "POP" is to estimate the storage requirements of the stiffness matrix before actually determining it. This information can be generated from the element connections with the nodes. For example, if an element connects 4 nodes, and if each node has 3 degrees of freedom in the global coordinate system, then the stiffness matrix of the element would be of dimension 12 x 12. This matrix can be partitioned four ways, in both row and column directions as shown in Fig. 6. The location of these sixteen submatrices in the total stiffness matrix can be determined by the address of the nodes to which the element is connected. If the element is connected to the nodes MA, MB, MC, and MD, then the addresses of the element submatrices in the total stiffness matrix are shown in Fig. 6.

If all the elements are connected to all the nodes, then the stiffness matrix of the structure will be fully populated. The non-zero elements in the matrix are considered as population. Since most of the elements connect only a few nodes, the stiffness matrices are usually sparsely populated. Determining the profile of the stiffness matrix population is the essential function of the routine "POP".

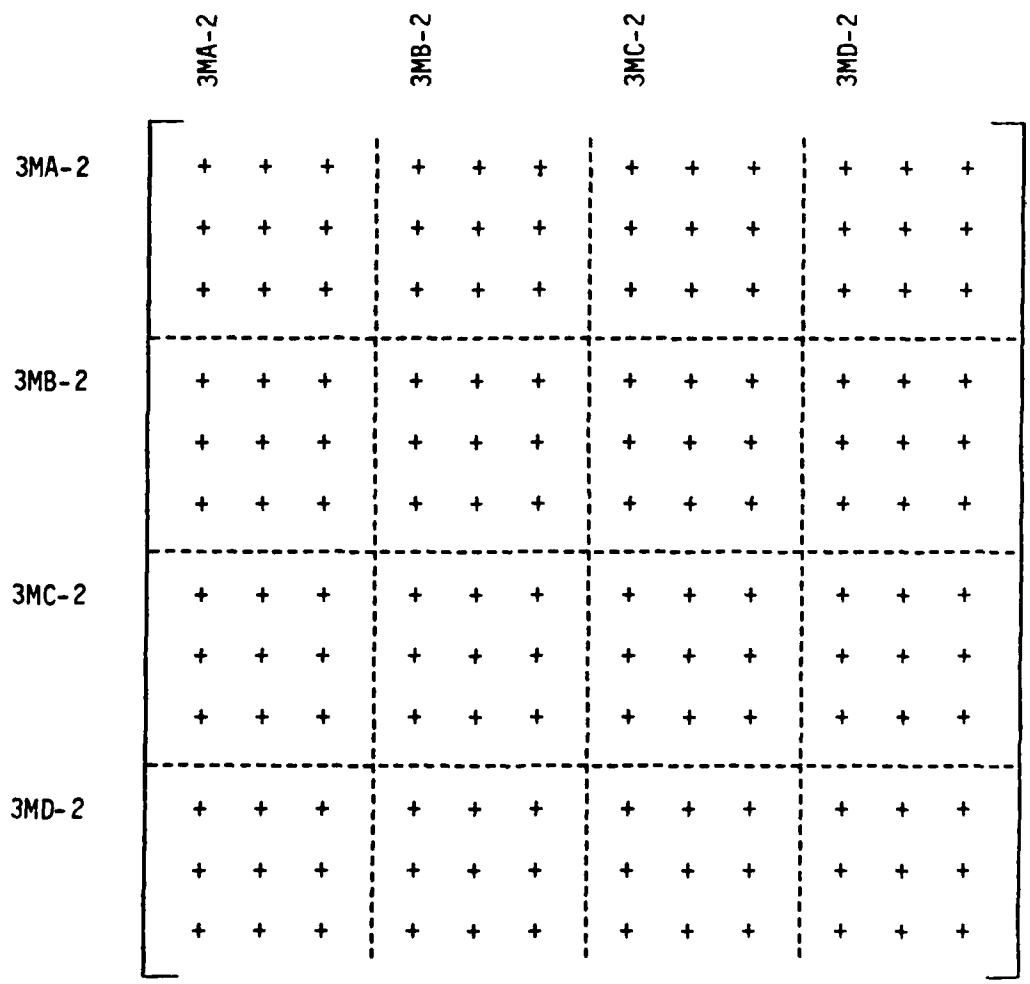


Fig. 6 Partitioned Element Stiffness Matrix and Addresses in the Total Stiffness Matrix

The distribution of the nonzero elements is dependent upon the way the nodes of the finite element model are numbered. Because of the symmetry of the stiffness matrix, only the lower or upper triangular matrix is considered. For the purpose of this discussion definitions of the following terms are in order. The gross population (P_{gross}) of the stiffness matrix is defined as the total number of elements in the upper triangle of the matrix. The net population (P_{net}) is the total number of non-zero elements in the upper triangle. Zeros resulting from transformations are not excluded from the net population. The apparent population ($P_{apparent}$) is the actual number of elements considered as nonzeros by a given solution scheme. From these definitions

$$P_{net} \leq P_{apparent} \leq P_{gross} \tag{92}$$

For a given structure P_{gross} and P_{net} are invariant and are given by

$$P_{gross} = \frac{N\,(N+1)}{2} \tag{93}$$

and

$$P_{net} = \frac{n\,(n+1)}{2}\,(\text{number of nodes}) + \sum_{i=1}^{m} \frac{n^2[k_i(k_i-1)]}{2} - n^2(NR) \tag{94}$$

where N is the total number of degrees of freedom of the structure, n is the number of degrees of freedom of each node (all the nodes are assumed to have the same number of degrees of freedom; when this is not true the necessary modification is simple), k_i is the number of nodes to which the i^{th} element is connected, and m is the number of elements in the structure. The quantity NR is given by

$$NR = \sum_{i=1}^{p} (b_i - 1) \tag{95}$$

where b_i is the number of elements connecting the same pair of nodes and p is the total number of pairs of directly connected nodes. If the structure consists of bar and/or beam elements only, NR is zero. For the example shown in Figure 6a, the value of NR is 3.

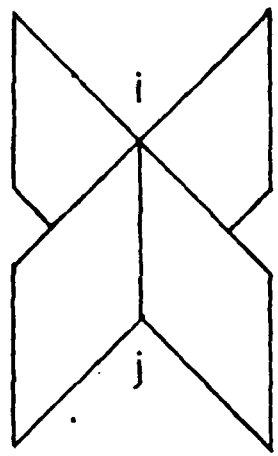
The quantity $P_{apparent}$ is dependent on the nature of the solution scheme used. For Gaussian elimination with no pivoting (LDL^T), $P_{apparent}$ may be defined as

$$P_{apparent} = \sum_{j=1}^{N} Q_j \tag{96}$$

where $Q_j = j - R_j + 1$ and where R_j is the row number of the first nonzero element in the j^{th} column. The solution scheme is most efficient when $P_{apparent} = P_{net}$. However, in large practical structures this condition is difficult to attain.

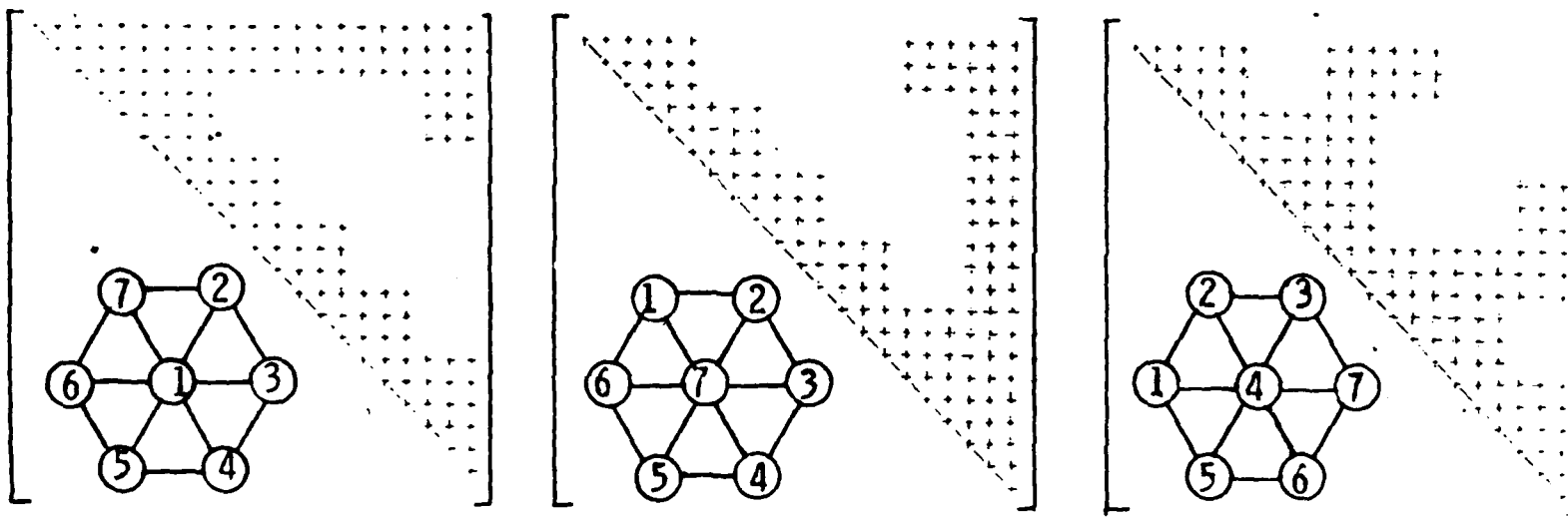
The value of $P_{apparent}$ changes with the node numbering scheme of the finite element model. The example shown in Figure 7 illustrates this point. A seven node three dimensional bar structure (n = 3) is numbered in three different ways and the resulting effect on the respective stiffness matrices is shown. The non-zero elements are marked by (+). The populations for the three cases are also given in the same figure. $P_{apparent}$ represents the number of storage locations required for the stiffness matrix.

a. INTERSECTING PLATES



NR = 3

SCHEME NO.	P_{GROSS}	P_{NET}	$P_{APPARENT}$
1	231	150	231
2	231	150	177
3	231	150	177



SCHEME 1 SCHEME 2 SCHEME 3

FIGURE 7: DISTRIBUTION OF NONZERO ELEMENTS IN THE STIFFNESS MATRIX

Subroutine "ELSTIC"

This routine generates the 3 x 3 elastic constants matrix for a given material (see Eq. 3).

Subroutine "COORD"

This routine establishes the local coordinate system for all the elements and also determines the nodal coordinates in the local system. It generates the direction cosine matrix which will be used to transform the element stiffness matrices to the global coordinate system (see Eqs. 13 and 16).

i. Bar Element

The local coordinate system of the bar element is established by drawing a line between the two nodes MA and MB (see Fig. 2) connecting the bar. The direction cosines are determined by

$$X_{Comp} = X_{MA} - X_{MB}$$

$$Y_{Comp} = Y_{MA} - Y_{MB} \qquad (97)$$

$$Z_{Comp} = Z_{MA} - Z_{MB}$$

$$L = (X_{Comp}^2 + Y_{Comp}^2 + Z_{Comp}^2)^{1/2} \qquad (98)$$

$$\ell_1 = \frac{X_{Comp}}{L} \quad m_1 = \frac{Y_{Comp}}{L} \quad n_1 = \frac{Z_{Comp}}{L} \qquad (99)$$

where X_{MA}, Y_{MA} and Z_{MA} are the three coordinates of the node MA in global coordinate system. The direction cosines ℓ_1, m_1, and n_1 become the first row of the 3 x 3 matrix $\underset{\sim}{A}$.

ii. Triangular Membrane Element

The local coordinate system of the triangular membrane element is established by assigning the local x-axis to the line joining nodes MA and MB. The direction cosines of this line are determined as in the case of the bar element. The plane of the plate is established by two unit vectors in the directions of the lines joining nodes MA-MB and MA-MC. If $\hat{a}$ and $\hat{b}$ are these two unit vectors, then the normal to the plane is obtained by

$$\hat{a} \times \hat{b} = \vec{c} \tag{100}$$

Since $\hat{a}$ and $\hat{b}$ are not orthogonal vectors, $\vec{c}$ is not a unit vector. The unit vector in this direction is given by

$$\hat{c} = \frac{\vec{c}}{|\vec{c}|} \tag{101}$$

The local z-axis is in the direction of the unit vector $\hat{c}$. Now the local y-axis is established by

$$\hat{c} \times \hat{a} = \hat{d} \tag{102}$$

The direction cosines of x and y become the first two rows of matrix $\underset{\sim}{A}$.

iii. Quadrilateral Membrane and Shear Panel

The local coordinate system of the quadrilateral membrane and the shear panel are established by a procedure similar to that of the triangle. The plane of the triangle connecting the three nodes MA, MB, and MC becomes the reference plane. Any warping in the quadrilaterals and shear panels is ignored. If there is too much warping in the quadrilaterals, it is better to divide them into two or more triangles or reduce the mesh size. In the case of excessively warped shear panels, the size of the grid must be

reduced. "OPTSTAT" does not have a provision for determining the warp and the consequent kick forces.

The node MA of the element becomes the origin of the element local coordinate system and the coordinates of the remaining nodes are determined by expressions similar to the following:

$$x_3 = (X_{MC} - X_{MA})\ell_1 + (Y_{MC} - Y_{MA})m_1 + (Z_{MC} - Z_{MA})n_1$$

$$y_3 = (X_{MC} - X_{MA})\ell_2 + (Y_{MC} - Y_{MA})m_2 + (Z_{MC} - Z_{MA})n_2$$

This subroutine also determines the coordinates of the fictitious node needed to break the quadrilateral and shear panels into four triangles. This interior node is established by

$$x_5 = \frac{x_1 + x_2 + x_3 + x_4}{4}$$

$$y_5 = \frac{y_1 + y_2 + y_3 + y_4}{4} \qquad (103)$$

where x_1, x_2 ....x_5 and y_1, y_2 ....y_5 are the coordinates of the five nodes (including the fictitious interior node) of the quadrilaterals and shear panels in the local coordinate system.

Subroutine "ELSTIF"

This subroutine determines the stiffness matrix of the bar by Eq. 22. It also transforms the bar stiffness matrix to the global coordinate system by

$$\underset{\sim}{K}_i = \underset{\sim}{a}_i^t \; \underset{\sim}{k}_i \; \underset{\sim}{a}_i \qquad (104)$$

Subroutines "PLSTIF" and "CRAMER"

The routine "PLSTIF" determines the element stiffness matrix of the triangle in the local coordinate system. This is also the basic routine for determining the stiffness matrices of the four triangles of the quadrilateral and the shear panel.

"PLSTIF" first calls the routine "CRAMER", which determines the inverse of the matrix $\underset{\sim}{X}$ by Cramer's rule. The matrix $\underset{\sim}{X}$ is given by Eq. 34. The determinant of $\underset{\sim}{X}$ represents twice the area of the triangle.

Then the "PLSTIF" subroutine determines the element stiffness matrix by Eq. 40. In determining the matrices $\underset{\sim}{\varepsilon}^{(i)}$ and $\underset{\sim}{\varepsilon}^{(j)}$, it takes advantage of the fact that the columns of Z^{-1} (see Eq. 33) represent unit displacement modes (see explanation under Eq. 34).

In computing the stiffness matrices of the triangles of the shear panels, "PLSTIF" considers only the shear strain energy. For example, in such a case, Eq. 40 becomes

$$\underset{\sim}{k} = \frac{1}{2}\left|\underset{\sim}{X}\right| t \begin{bmatrix} \varepsilon_{xy}^{(1)}G\varepsilon_{xy}^{(1)} & \varepsilon_{xy}^{(1)}G\varepsilon_{xy}^{(2)} & \cdots & \varepsilon_{xy}^{(1)}G\varepsilon_{xy}^{(6)} \\ \vdots & & & \vdots \\ \varepsilon_{xy}^{(6)}G\varepsilon_{xy}^{(1)} & \varepsilon_{xy}^{(6)}G\varepsilon_{xy}^{(2)} & \cdots & \varepsilon_{xy}^{(6)}G\varepsilon_{xy}^{(6)} \end{bmatrix} \qquad (105)$$

Subroutine "QDRLTL"

This subroutine simply manages the routines "PLSTIF", "SUM", and "CONDNS" in computing the stiffness matrix of the quadrilateral membrane and shear panel. This routine also makes provision for assigning different sides as reference axis for the shear panels.

Subroutine "SUM"

This subroutine adds the four triangle stiffness matrices computed by "PLSTIF" to produce a 10 x 10 stiffness matrix (including two degrees of freedom for the interior node) for the quadrilateral or shear panel.

Subroutine "CONDNS"

This routine condenses the 10 x 10 quadrilateral or shear panel stiffness matrix to an 8 x 8 matrix. The condensation is done by using Eq. 56.

Subroutine "CHANGE"

This routine interchanges the rows and columns of the quadrilateral (or shear panel) stiffness matrix so that the element degrees of freedom are in ascending order before addition to the structure stiffness matrix. This step is necessary because the routine "ASEMBL" assumes that the element degrees of freedom are in ascending order.

Subroutine "TRNSFM"

This routine transforms the plate element stiffness matrices from the local to the global coordinate system by (see Eq. 16)

$$\underset{\sim}{K}_i = \underset{\sim}{a}_i^t \; \underset{\sim}{k}_i \; \underset{\sim}{a}_i \tag{106}$$

where $\underset{\sim}{K}_i$ is the transformed element stiffness matrix of the i^{th} element.

Subroutine "ASEMBL"

This routine adds the element stiffness matrices to the total stiffness matrix.

$$\underset{\sim}{K} = \sum_{i=1}^{m} \underset{\sim}{K}_i \tag{107}$$

For an explanation of the rules of this addition see the description of subroutine "POP". It should be noted that only the upper half of the stiffness matrix is stored. This storage is columnwise starting with the first non-zero element above the diagonal.

Subroutine "PRINTK"

The purpose of this routine is to print the stiffness matrix (if desired) rowwise starting with the first non-zero element and proceeding to the diagonal.

Subroutine "BOUND2"

This routine eliminates the rows and columns corresponding to the constrained degrees of freedom and condenses the stiffness matrix.

Subroutine "REDUCE"

This routine eliminates the rows of the applied force matrix corresponding to the constrained degrees of freedom. It is assumed that each column of the force matrix represents an independent load condition.

Subroutine "GAUSS"

"GAUSS" solves the load deflection equations (Eq. 17) by Gaussian elimination. The first step of the solution is the decomposition of the stiffness matrix by Eq. 18. The next two steps represent forward and back substitution using Eqs. 19 and 20 respectively. For the solution of additional load vectors only the steps FBS have to be repeated. If "GAUSS" is entered with any value other than 0 for the parameter NDCOMP, only the last two steps will be executed. The matrices $\underset{\sim}{L}$ and $\underset{\sim}{D}$ are stored in place of the original stiffness matrix.

Subroutine "RESTOR"

This routine restores the displacement or force matrix to full size by assigning zero values to boundary degrees of freedom.

Subroutine "ELFORC"

This routine extracts the element displacements from the global coordinate system and transforms them to the local coordinate system by Eq. 13.

Subroutine "STRESS"

The purpose of the "STRESS" routine is to compute strains and stresses in the triangular element. It first calls the routine "CRAMER" which computes $\underset{\sim}{X}^{-1}$ (Eq. 34) by Cramer's rule. The strains in the element are then calculated by Eqs. 30 and 35 thru 37. The stresses in the element are computed by Eq. 2. Also it computes the strain energy and the effective stress in the element by Eqs. 1 and 45 respectively.

Subroutine "QLSTRS"

This routine prepares the data for computing stresses in the four triangles of the quadrilateral or shear panel elements. First it determines the interior node displacements from the corner node displacements using Eq. 54. Then it calls subroutine "STRESS" to compute the stresses in the four triangles. It adds the strain energy of the four triangles to obtain the total strain energy. It identifies the triangle with the largest effective stress and normalizes the effective stress of the three remaining triangles with respect to this largest value.

Subroutine "PRNTDR"

This subroutine prints out the table of node information. This includes the node number, its coordinates, applied forces, and the displacements.

NAME	NUMBER OF CARDS	CALLED FROM
OPTSTAT	895	Main Program
POP	62	OPTSTAT
ELSTIC	15	OPTSTAT
COORD	44	OPTSTAT
ELSTIF	21	OPTSTAT
PLSTIF	46	OPTSTAT, QDRLTL
CRAMER	19	PLSTIF, STRESS
QDRLTL	32	OPTSTAT
SUM	23	QDRLTL, QLSTRS
CONDNS	36	QDRLTL, QLSTRA
CHANGE	25	CONDNS
TRNSFM	36	OPTSTAT
ASEMBL	41	OPTSTAT
PRINTK	15	OPTSTAT
BOUND2	35	OPTSTAT
REDUCE	18	OPTSTAT
GAUSS	57	OPTSTAT
RESTOR	28	OPTSTAT
ELFORC	22	OPTSTAT
STRESS	33	OPTSTAT, QLSTRS
QLSTRS	65	OPTSTAT
PRNTOR	39	OPTSTAT
PREPAR	60	OPTSTAT
TRECON	40	OPTSTAT
GAUSS1	35	OPTSTAT
UNITEG	40	OPTSTAT
LMSIZE	45	OPTSTAT
LAYCALC	60	OPTSTAT
LAYPR	28	OPTSTAT
TOTAL	1911	

Table 1: Program Description

7. INPUT INSTRUCTIONS

Input for the programs is divided into a number of card sets. Each card set will consist of one or more cards. Only three Formats are used for input. An integer Format (1415), a floating point Format (6F10.0) and a mixed Format 3(F10.0,2I5). The first five card sets will each have one card regardless of the size of the problem. The number of cards required for the remaining card sets depends on the problem size. The first card set indicates the number of problems (structures) to be analyzed. If this number is more than one, the program assumes that the remaining card sets will be supplied for each problem one after the other. The next card set is for the title of the problem. Card sets three and four define the basic parameters like the number of elements, nodes etc. And set five defines minimum size etc. The remaining card sets define material properties (6-11), type of elements (12), element connections (13, 14, 15, 16), material code for the elements (17 and 18) etc. The input instructions in the following pages explain the function of each card set.

INPUT FOR PROGRAM OPTSTAT

CARD SET (FORMAT)	PARAMETER		DESCRIPTION
1 (14I5)	NSTR		Number of problems to be solved.
2 (8A10)	TITLE		A user selected title of the problem to be solved (alpha-numeric description).
	Card sets 3, 4 and 5 (each contains only one card) define a set of control parameters to provide flexibility to the user in defining the problem and selecting the input (output) options.		
3 (14I5)	MEMBS		Number of elements.
	JOINTS		Number of nodes.
	NBNDRY		Number of restrained degrees of freedom.
	LOADS		Number of loading conditions.
	MM		MM [= 2 Two dimensional problem = 3 Three dimensional problem
	LMTDSP	LMTDSP	[<1 No displacement constraints =1 Displacement constraint is the same for all nodes. >1 Displacement constraint can vary per node.
	LMTCCL		Number of cycles of iteration using the recursion relation based on displacement gradients.
	INCHES	INCHES	[=1 Coordinate data is in inches. ≠1 Coordinate data is in feet.
	KIPS	KIPS	[=1 Applied forces are in kips. ≠1 Applied forces are in pounds.
	LSTCCL		Number of cycles of iteration using the recursion relation based on the energy stored in each element.
	NR		Variable used only for calculating the net population of the stiffness matrix. It has no other role in the program. Thus if the net population figure is of little interest, any arbitrary number may be input.

CARD SET (FORMAT)	PARAMETER		DESCRIPTION
	IAREAS	IAREAS	=1 Input initial thicknesses of the elements ≠1 Initial thicknesses are set by the program. (1.0 in.)
	INSIST	INSIST	=0 Design in the strength mode until the weight increases and then either quit or proceed to the displacement mode. =1 Complete all cycles in the strength mode and proceed to the displacement mode. =2 Directly proceed to the displacement mode.
	LPRINT	LPRINT	=0 No additional output requested for layered composite elements. ≠0 Additional output for layered composite elements.
4 (14I5)	NMAT		Total number of materials (isotropic + composite).
	NISOTR		Number of composite materials.
	INDANG	INDANG	=0 For a layered composite element, the 0° fibers are defined per element with respect to the global coordinate system. =1 For a layered composite element, the 0° fibers are defined per element with respect to the local element coordinate system. =2 The direction cosines of the 0° fibers are defined with respect to the global coordinate system.
	LAYERD	LAYERD	=0 Problem contains no layered composite elements. =1 Problem contains layered composite elements.
	NCDPEL	NCDPEL	=1 Element data is read one card per element. ≠1 Element data is read in condensed format.
	NCDPND	NCDPND	=1 Node data is read one card per node. ≠1 Node data is read in condensed format.
	INDMIN	INDMIN	=0 Minimum allowable size in the same for all elements. =1 Minimum sizes of the elements are input.

CARD SET (FORMAT)	PARAMETER		DESCRIPTION
	KANLYZE	KANLYZE	=0 Use the program for structural optimization. =1 Use the program for structural analysis only. No resizing.
	MAXSZE	MAXSZE	=0 No maximum size will be specified for the elements. =1 Maximum allowable sizes of the elements are input.
	MNLAYR	MNLAYR	=1 Minimum proportions of 0°, 90°, ±45° layers will be input for each member. ≠1 Minimum proportions of the 0°, 90°, ±45° layers are the same for all elements.
5 (6F10.3)	AEMNMM		Minimum allowable element size.
	DINCR		A parameter to determine the active set of displacement constraints. Usually $1.01 \leq DINC \leq 1.1$
	THKLAM		Minimum layer thickness.
	SPRDF		Shear panel reduction factor. Usually $.5 \leq SPRDF \leq .8$

Material Properties Data: Card Sets 6 thru 11 are for defining material properties data.

6 (6F10.3)	YOUNGM(I)	Youngs modulus in psi/10^6 of the I^{th} material.
	POISON(I)	Poisson's ratio of the I^{th} material.
	RHO1(I)	Density in lbs/in^3 of the I^{th} material.
	I = 1,..., NMAT	

Card Sets 7 thru 10 are relevant only if anisotropic materials are used. They should be skipped if NISOTR = 0 (See Card Set 4 for the definition of NISOTR).

CARD SET (FORMAT)	PARAMETER	DESCRIPTION
7 (6F10.3)	ELCNST(I)	Elastic modulus in psi/10^6 transverse to the fiber direction for the I^{th} composite material.
	ELCNST(I+1)	Shear modulus in psi/10^6 for the I^{th} composite material.
	I = 1, 2 * NISOTR, 2	

Card Sets 8 thru 10 are for defining the orientation of the anisotropic material property axis. The user should select one of these three options based on the value of the parameter "INDANG" in Card Set 4.

8 (6F10.3)	XANG(I)	The angle in degrees that the 0° fibers of the I^{th} element makes with the local element coordinate system.
	I = 1,..., MEMBS	
9 (6F10.3)	XANG(I)	The angle in degrees that the 0° fibers of the I^{th} element makes with the X-axis of the global coordinate system.
	YANG(I)	The angle in degrees that the 0° fibers of the I^{th} element makes with the Y-axis of the global coordinate system.
	ZANG(I)	The angle in degrees that the 0° fibers of the I^{th} element makes with the Z-axis of the global coordinate system.
	I = 1,..., MEMBS	
10 (6F10.3)	AX	Direction cosine of the angle the 0° fibers make with the X-axis of the global coordinate system.
	AY	Direction cosine of the angle the 0° fibers make with the Y-axis of the global coordinate system.
	AZ	Direction cosine of the angle the 0° fibers make with the Z-axis of the global coordinate system.

CARD SET (FORMAT)	PARAMETER	DESCRIPTION
11 (6F10.3)	ALSTRS(I)	Tension allowable of the I^{th} material in psi/10^3 parallel to the 0° fiber direction.
	ALSTRS(I+1)	Compression allowable of the I^{th} material in psi/10^3 parallel to the 0° fiber direction.
	ALSTRS(I+2)	Tension allowable of the I^{th} material in psi/10^3 transverse to the 0° fiber direction.
	ALSTRS(I+3)	Compression allowable of the I^{th} material in psi/10^3 transverse to the 0° fiber direction.
	ALSTRS(I+4)	Shear allowable of the I^{th} material in psi/10^3 transverse to the 0° fiber direction.
	I = 1, 5*NMAT, 5	

Card Sets 12 thru 23 define element types, connections, material code and properties. The user can choose either a condensed form or a card per element form by giving NCDPEL=0 or 1 in Card Set 4. Card Sets 12 thru 22 describe the condensed form. Card Set 23 describes the alternate form. The user should choose either one or the other but not both.

ELEMENT TYPES

12 (14I5)	NNODES(I) I = 1,..., MEMBS	Element Type NNODES(I) =2 BAR =3 TRIANGLE =4 QUADRILATERAL MEMBRANE =5 SHEAR PANEL

ELEMENT CONNECTIONS

13 (14I5)	MA(I) I = 1,..., MEMBS	First node number of each element.

CARD SET (FORMAT)	PARAMETER	DESCRIPTION
14 (14I5)	MB(I) I = 1,..., MEMBS	Second node number of each element.
15 (14I5)	MC(I) I = 1,..., MEMBS	Third node number of each element.
16 (14I5)	MD(I) I = 1,..., MEMBS	Fourth node number of each element.

NOTE: For bars leave MC(I) and MD(I) blank. For triangles leave MD(I) blank. For each element let MA(I) be the lowest node number and MB(I) be the next lowest. For Quadrilaterals and Shear Panels, MC(I) and MD(I) are determined by continuing in the direction defined by MA(I) and MB(I).

MATERIALS CODE FOR THE ELEMENTS

Card Set 17 is relevant only when there are two or more materials: i.e. IF NMAT>1 in Card Set 4

17 (14I5)	MYOUNG(I) I = 1,..., MEMBS	Material property number of the I^{th} element.

Card Set 18 is relevant only for layered composite materials. IF LAYERD=0 in Card Set 4, skip Card Set 18.

18 (14I5)	LAM(I) I = 1,..., MEMBS	LAM(I) =0 Isotropic element. =1 Fiber orientations 0°, 90°, ±45° are in the proportions .25, .25, .50 =2 Fiber orientations 0°, 90° are in the proportions .50, .50 =3 Fiber orientations ±45° are in the proportions 1.00 =4 Fiber orientations 0°, ±45° are in the proportions 1/3, 2/3. >4 Fiber orientations 90°, ±45° are in the proportions 1/3, 2/3.

CARD SET (FORMAT)	PARAMETER	DESCRIPTION
	ELEMENT SIZES	
	Card Sets 19 thru 21 are necessary only if the user wants to give initial sizes for the elements. Otherwise the program assigns equal sizes for all the elements. The parameter IAREAS (0 or 1) in Card Set 3 indicates the choice. IF IAREAS=0, skip Card Sets 19 thru 21.	
19 (6F10.3)	AE(I) I = 1,..., MEMBS	Initial thickness of each element. For a bar, thickness is cross-sectional area.
	If all the elements are made of isotropic materials, skip Card Sets 20 and 21. Check the parameter LAYERD in Card Set 4.	
20 (6F10.3)	AEX(I) I = 1,..., MEMBS	Proportion of fibers in the 0° direction for the I^{th} element.
21 (6F10.3)	AEY(I) I = 1,..., MEMBS	Proportion of fibers in the 90° direction for the I^{th} element.
	Card Set 22 is necessary only if there are individual minimum sizes for the elements. If the minimum size is the same for all the elements, then it is defined on Card Set 5 as AEMNMM. IF INDMIN=0, (Card Set 4) skip Card Set 22.	
22 (6F10.3)	AEMNM(I) I = 1,..., MEMBS	Minimum size of the I^{th} element.
	Card Set 23 is an alternate form for element information, and it is selected by the user when NCDPEL=1 (Card Set 4).	

CARD SET (FORMAT)	PARAMETER	DESCRIPTION
23 (8I5, 4F10.3)	KX	Element number.
	NNODES(I)	See CARD SET 12
	MYOUNG(I)	See CARD SET 17
	MA(I)	See CARD SET 13
	MB(I)	See CARD SET 14
	MC(I)	See CARD SET 15
	MD(I)	See CARD SET 16
	LAM(I)	See CARD SET 18
	AE(I)	See CARD SET 19
	AEX(I)	See CARD SET 20
	AEY(I)	See CARD SET 21
	AEMNM(I)	See CARD SET 22
	I = 1,..., MEMBS	

Card Set 24 is necessary only when there are maximum limits on the element sizes. If the parameter MAXSZE=0 (Card Set 4), skip Card Set 24.

24 (6F10.3)	AEMAX(I) I = 1,..., MEMBS	Maximum size of the I^{th} element.

Card Set 25 is relevant only for layered composite materials. If MNLAYR≠1 in Card Set 4, skip Card Set 25.

25 (6F10.3)	AEXMIN(I)	Minimum proportion of 0° layers for the I^{th} element.
	AEYMIN(I)	Minimum proportion of 90° layers for the I^{th} element.
	AEXYMIN(I)	Minimum proportion of ±45° layers for the I^{th} element.
	I = 1,..., MEMBS	

CARD SET (FORMAT)	PARAMETER	DESCRIPTION

Card Sets 26 and 27 define grid point coordinates. The user can choose either a condensed form or a card per grid point form by giving NCDPND=0 or 1 in Card Set 4. Card Set 26 represents the condensed form and Card Set 27 the alternate form. The user should choose one or the other but not both.

26 (6F10.3)	X(I)	X coordinate of the I^{th} node.
	Y(I)	Y coordinate of the I^{th} node.
	Z(I)	Z coordinate of the I^{th} node.
	I = 1,..., JOINTS	

NOTE: For MM = 2, Z(I) is not input.

27 (I5, 3F10.0)	KX	Node Number
	X(I)	
	Y(I)	See CARD SET 26
	Z(I)	
	I = 1,..., JOINTS	

Card Set 28 is for defining the boundary degrees of freedom.

28 (14I5)	IBND(I) I = 1,..., NBNDRY	Degree of freedom numbers of those nodes which are restrained. For node K the degree of freedom numbers are 3*K-2, 3*K-1, and 3*K for MM=3 and 2*K-1, 2*K for MM=2.

CARD SET (FORMAT)	PARAMETER	DESCRIPTION
	Card Sets 29 and 30 define the loading on the structure.	
29 (14I5)	NJLODS(I) I = 1,..., LOADS	Number of load components in the I^{th} loading condition.
30 3(F10.0, 2I5)	TFR(J)	Value of the load.
	IM(J)	Direction of the load. IM(J) =1 x direction. =2 y direction. =3 z direction.
	JM(J) J = 1,..., NJLODS(I)	Number of the node where the load is applied.
	Card Sets 31 thru 33 define the displacement constraints on the structure. The options for displacement constraints are defined by the parameter LMTDSP on Card Set 3. IF LMTDSP≤0, skip Card Sets 31 thru 33. IF LMTDSP=1, use only Card Set 31. IF LMTDSP≥1, use Card Sets 32 and 33.	
31 (6F10.3)	DEFMAX(J) J = 1,..., MM	Absolute value of the displacement constraint in the j^{th} direction for all nodes. J =1 x direction. =2 y direction. =3 z direction.
32 (14I5)	NLTDEF	Number of displacement constraints.
33 3(F10.0, 2I5)	TFR(I)	Magnitude of the displacement constraint.
	IM(I)	Direction in which the constraint is applied. IM(I) =1 x direction. =2 y direction. =3 z direction.
	JM(I) I = 1,..., NLTDEF	Number of the node where the constraint is applied.

Output for Program OPTSTAT

Output for Program OPTSTAT consists of the following:

1) Untitled echo of CARD SETS 2, 3, 4 and 5.

2) Materials Table from CARD SETS 6, 7 and 11.

3) Element Table from CARD SETS 8-10, 12-23.

4) Untitled echo of CARD SETS 26 and 27.

5) Boundary data, i.e. contents of array IBND (CARD SET 29).

6) Summary of Applied Loads Table.

7) Output from Subroutine POP concerning the distribution of elements in the stiffness matrix. This information is generated before the stiffness matrix of the structure is assembled.

(a) Gross Population = total number of elements in the upper triangle of the matrix.

Net Population = actual population of possible non-zero elements in the upper triangle of the stiffness matrix. This number would be correct only if NR is correct in CARD SET 2.

Apparent Population = actual number of elements considered as non-zero by a given solution scheme. Thus the apparent population represents the number of storage locations required for the stiffness matrix.

(b) Starting Row Numbers for each column - the number of the row where the first non-zero element occurs in each column.

(c) Number of Diagonal Elements in Single Array Stiffness Matrix. For each Column I the actual number of elements, ID(I), in the upper triangular matrix up to and including that column, i.e.

$$ID(I) = \frac{I(I+3)}{2} - \sum_{j=1}^{I} b_j$$

where b_j is the row number given for Column I in (b). Thus for the last column, ILAST,

ID(ILAST) = Apparent Population

8) Initial sizes of the elements (CARD SET 19).

9) BASEAE - Scaling parameter based on the total energy in the structure.

BASEAE - Scaling parameter based on displacement constraints.

10) MEMB. NO. - Element number.

SCALING FACTOR - Maximum positive ratio of tension (compression) in the element to the tension (compression) allowable over all loading conditions if this ratio is >1.0.

11) Maximum effective stress ratio (if analysis only, i.e. KANLYZE = 1)

12) If maximum sizes of the elements are input, i.e. (MAXSZE = 1)

Scale Factors

DESIRED - Either BASEAE as given in 9) divided by 10^6 or (BASEAE/10^6)* the last scaling factor given in 10).

ACTUAL - Minimum ratio over all the elements of the maximum allowable size of the element to the relative size of the element which is $\leq$ desired scale factor.

RATIO - Desired scale factor/actual scale factor.

CRITICAL MEMBER - Element number from which the actual scale factor was calculated.

If critical member = 0, either there were no items output in 10) or no actual scale factor was computed, i.e. Desired scale factor = Actual scale factor.

13) BASE AE - Scaling parameter
Weight of the Structure
Weight of the Membrane Elements
Cycles in Search - Current number of cycles of iteration using the recursion relation based on displacement gradients.
Structure Number - Number of the current data set (CARD SET 1).
No. of Loads - Number of loading conditions.
Cycle No. - Total number of cycles of iteration.
Weight of the Shear Panels.
Weight of the Bar Elements.

14) STEP REDUCED
If the weight goes up in the displacement mode, the relative sizes of the elements are reduced.

15) NDUMMY - The number of times the deflection limits have been exceeded.

NUFR - The degree of freedom numbers where the deflection limits have been exceeded.

16) Relative Areas of Members - (Absolute thicknesses of the elements x Young's modulus in psi)/Scaling parameter.

Output 9) and 10) is repeated for each cycle.

17) Output for each element after the optimization is completed.

(a) MEMBER - Element Number

(b) THICK - Absolute thickness of the resized member.

(c) AREA - Area of the element. For a bar area is length.

(d) TYPE - Type of element (CARD SET 12).

(e) MA, MB, MC, MD - defined in CARD SETS 13, 14, 15, and 16

(f) SIGMA-X (σ_x), SIGMA-Y (σ_y), SIGMA-XY (σ_{xy})-
Stresses in the x-y local coordinates of the element.

(g) ESRATIO - Effective stress ratio in the element determined by the Von Mises Criterion.

The stress output varies per element type.
(i) BAR SIGMA-X only
(ii) TRIANGLE SIGMA-X, SIGMA-Y, SIGMA-XY
(iii) QUADRILATERAL MEMBRANE
The Quadrilateral membrane element is divided into 4 triangles for analysis. SIGMA-X, SIGMA-Y, SIGMA-XY are for that triangle with the maximum effective stress ratio. This maximum effective stress ratio is given by ESRATIO.
(iv) SHEAR PANEL
The Shear Panel is also divided into 4 triangles for analysis. SIGMA-XY (τ_{xy}) is for that triangle with the maximum effective stress ratio. This maximum effective stress ratio is given by ESRATIO.

For layered composite elements output (f) is replaced by
(i) (LAM) - The total number of layers.
(ii) (THKO) - Total thickness of the layers in the 0° fiber direction.
(iii) (AEX) - Proportion of fibers in the 0° direction.
(iv) (THK90) - Total thickness of the layers in the 90° fiber direction.
(v) (AEY) - Proportion of fibers in the 90° direction.

(h) ALS1 - Tension allowable of the element parallel to the 0° fiber direction.

ALS2 - The ratio of the compression allowable parallel to the 0° fiber direction to ALS1.

ALS3 - The ratio of the tension allowable transverse to the 0° fiber direction to ALS1.

ALS4 - The ratio of the compression allowable transverse to the 0° fiber direction to ALS1.

ALS5 - The ratio of the shear allowable transverse to the 0° fiber direction to ALS1 (SEE CARD SET 11).

(i) ENERGY - Strain Energy in the element.

NOTE: If the number of loading conditions is greater than 1, output (g) and (i) are given continuously for each load case.

18) The total energy for each loading condition.

19) Output for each node after the optimization is completed.

(a) JOINT - Node Number

(b) X, Y, Z - x, y, and z coordinate of the node.

(c) FORCE-X, FORCE-Y, FORCE-Z - applied forces in the x, y and z direction.

(d) DISPL-X, DISPL-Y, DISPL-Z - Displacements in the x, y and z direction.

NOTE: If the number of loading conditions is greater than 1, output (c) and (d) are given continuously for each load case.

If the problem contains layered composite elements, additional output can be requested (See CARD SET 3).

20) MEMB - Element No.

Total Number of Layers per element.

The number of layers in each of the fiber directions (0, 90, $\pm$45)

21) Based on the output in 14), AEX, AEY and THICK are recalculated and a structural analysis is performed. Output 10), 11), 12) and 13) are repeated.

Design Example

The three spar wing shown in Figure 6 is idealized by membrane quadrilaterals, shear panels and bars (axial force members). The top and bottom skins are graphite epoxy layered composite elements with 0°, 90° and ±45° fibers. The spars and ribs are idealized by aluminum shear panels. In addition, the top and bottom nodes are connected by bar elements or posts. The root section of the wing is assumed to be fixed. The wing is designed for two independent loads. These loading conditions are generated by simplified pressure distributions representative of a subsonic, forward-center-of-pressure loading and a supersonic near-uniform-pressure loading. The detailed distribution of the loading on the nodes is given in Table 1. The material properties of the graphite epoxy and aluminum are given in Table 2. The constraints are only on stresses and minimum sizes. The wing was optimized by OPTSTAT and ASOP 3[18]. The distribution of the composite layers and the thickness of the spars and webs are given in Figure 7. Figure 7a gives the composite layer distribution in the wing skins. The top figures were obtained by OPTSTAT and the bottom figures by ASOP 3. The details of the ply orientations in 0°, 90° and ±45° were given in Figure 7b. Figure 3c gives the material distribution in the substructure. The design obtained by OPTSTAT weighs approximately 34 lbs. The ASOP 3 wing was about 40 lbs. (See Figure 7c). There was substantial difference in the composite material distribution of wing skins obtained by the two programs. The design obtained by ASOP 3 is heavier and stiffer than that obtained by OPTSTAT. The difference in the two designs can be attributed to the

resizing algorithms, methods of calculation of stresses and the failure criteria in these two programs. The OPSTAT program resizes the elements by using as energy criterion, while ASOP 3 resizes by a stress ratio criterion. In addition there are differences in the way stresses are computed.

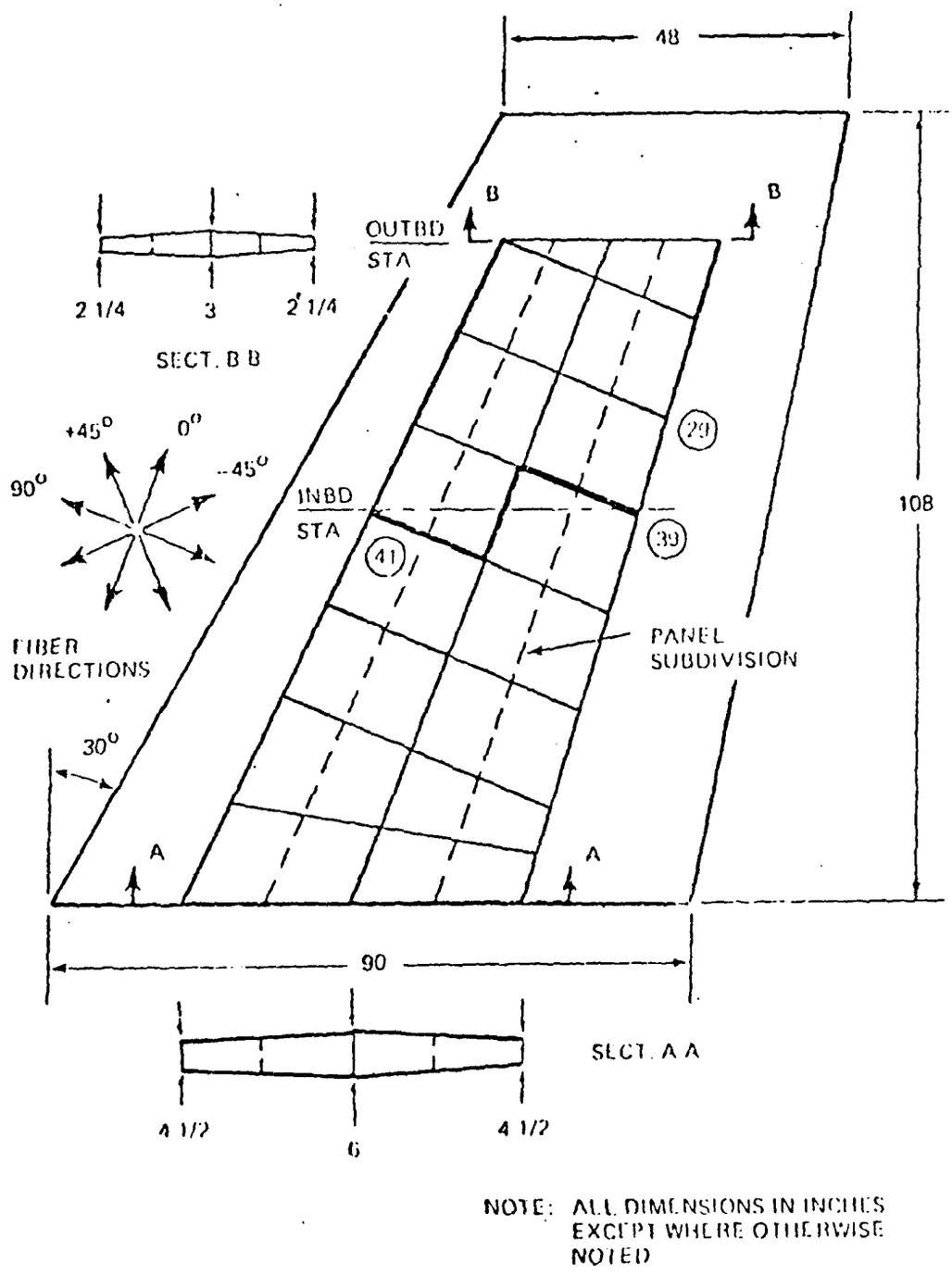


Figure 6. Aerodynamic Planform and Primary Structural Arrangement of Wing

Note: Top figures are from Program OPTSTAT; bottom figures ( ) are from Program ASOP 3.

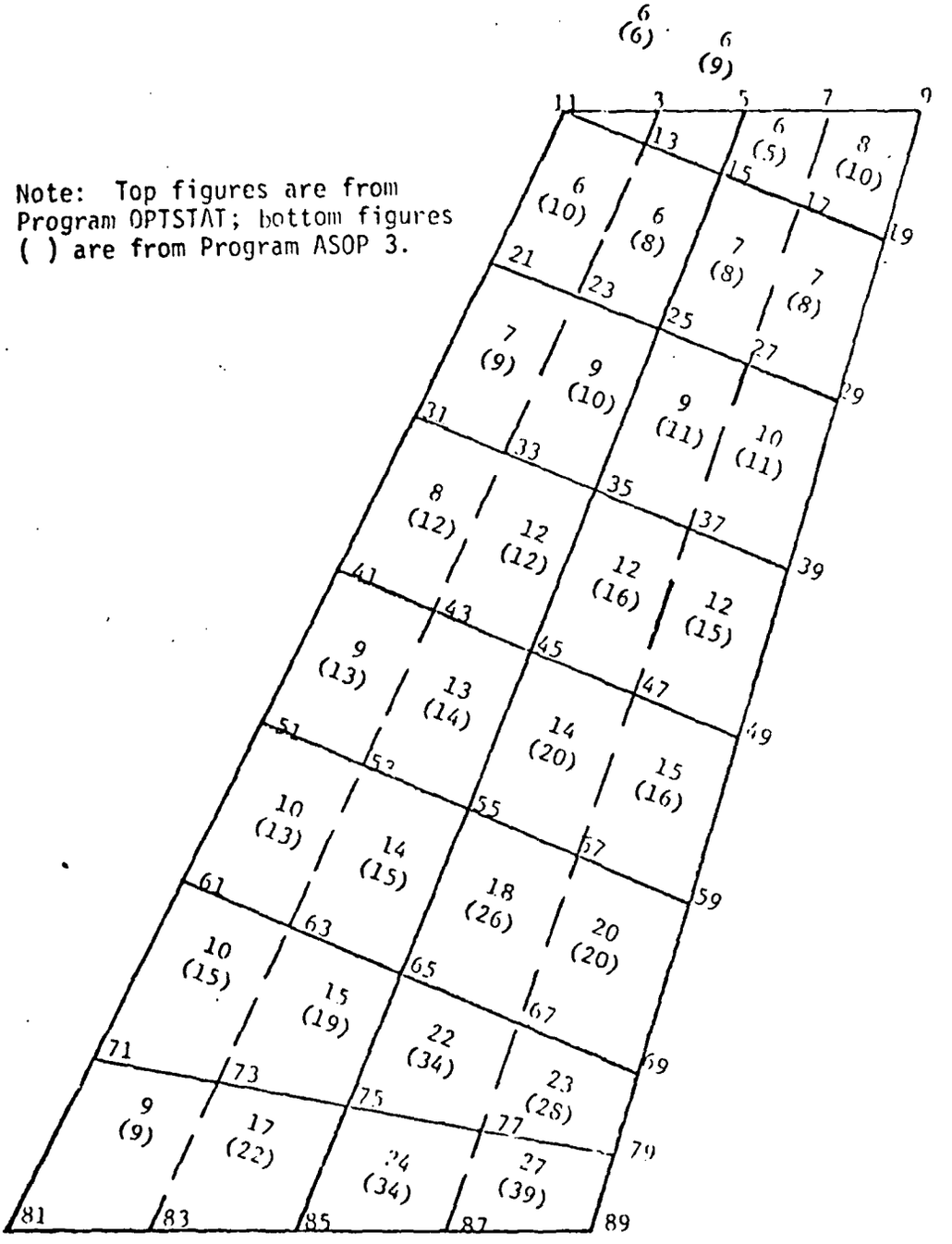


Figure 7a. Total Number of Layers in the Top Skin (Bottom skin is the same)

Note: Top figures are from Program OPTSTAT; bottom figures ( ) are from Program ASOP 3.

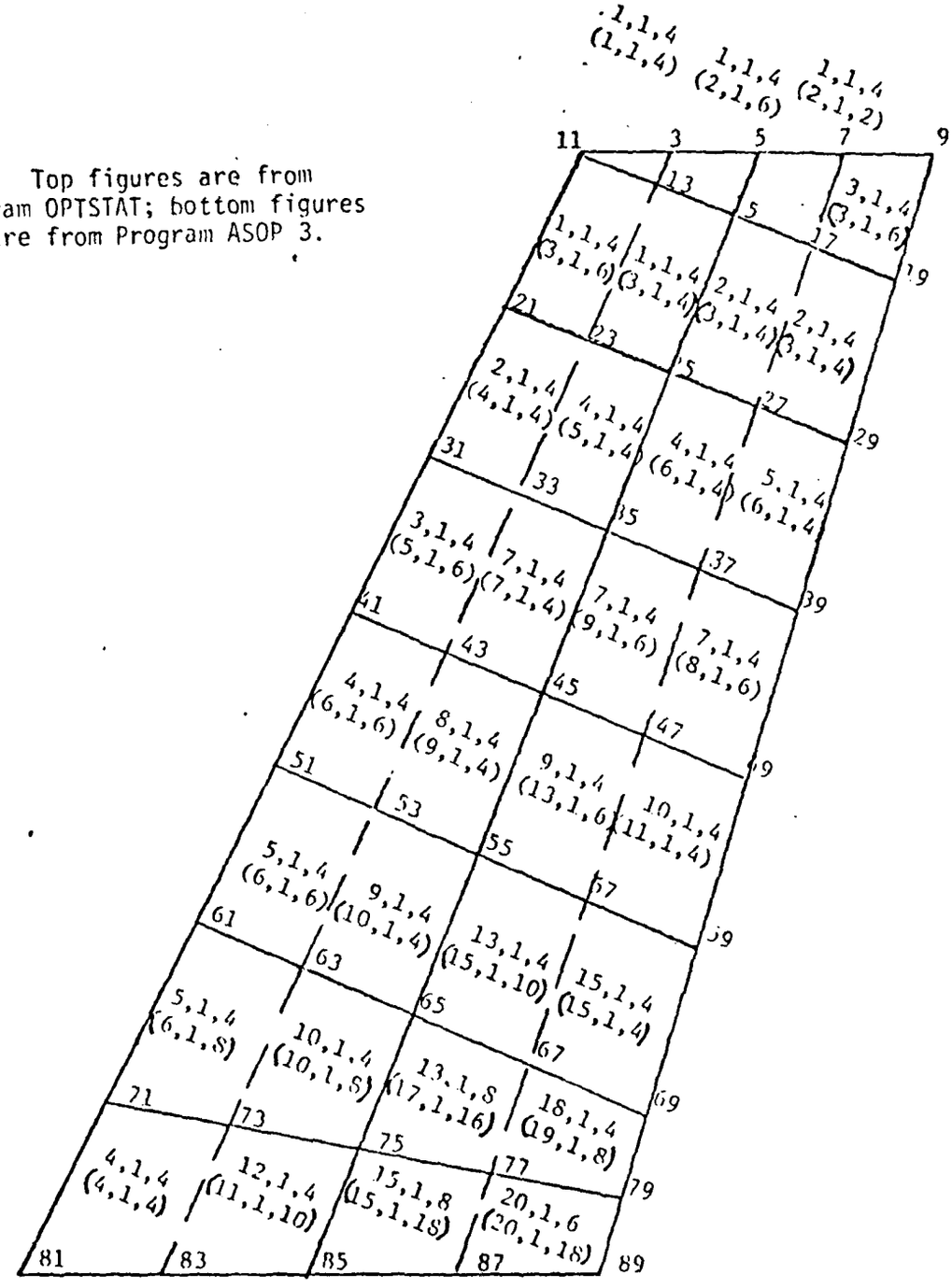


Figure 7b. Distribution of Fibers in 0°, 90°, ±45° Direction

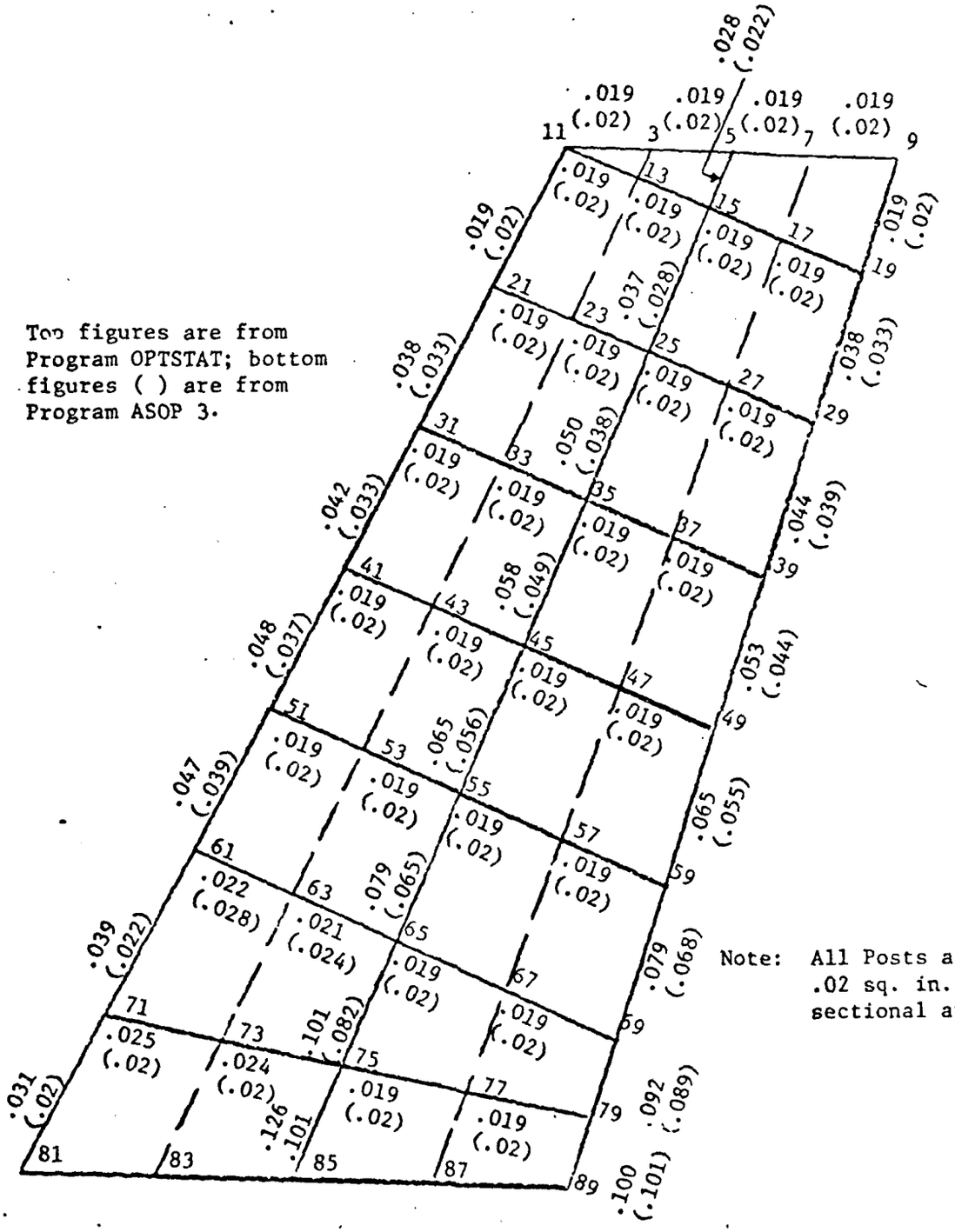


Figure 7c. Substructure Gages for the Wing

Table 1a. Loading Data

Node No.	x		y		z		Node No.	x		y		z	
	L.C.1	L.C.2	L.C.1	L.C.2	L.C.1	L.C.2		L.C.1	L.C.2	L.C.1	L.C.2	L.C.1	L.C.2
1	0	0	0	0	29	30	35	0	0	0	0	365	230
2	0	0	0	0	29	30	36	0	0	0	0	365	230
3	-2800	-2420	-6960	-6020	1130	979	37	-4250	-1270	1740	518	974	336
4	2800	2420	6960	6020	1130	979	38	4250	1270	-1740	-518	974	336
5	0	0	0	0	91	56	39	1820	4160	-743	-1700	694	1410
6	0	0	0	0	91	56	40	-1820	-4160	743	1700	694	1410
7	-9870	-4020	-9780	-3980	1130	474	41	0	0	0	0	365	402
8	9870	4020	9780	3980	1130	474	42	0	0	0	0	365	402
9	205	351	-7380	-12600	926	1530	43	0	0	0	0	378	313
10	-205	-351	7380	12600	926	1530	44	0	0	0	0	378	313
11	0	0	0	0	178	194	45	0	0	0	0	392	247
12	0	0	0	0	178	194	46	0	0	0	0	392	247
13	0	0	0	0	214	175	47	-4440	-1320	1820	541	1050	361
14	0	0	0	0	214	175	48	4440	1320	-1820	-541	1050	361
15	0	0	0	0	253	157	49	1890	4330	-773	-1770	742	1500
16	0	0	0	0	253	157	50	-1890	-4330	773	1770	742	1500
17	-5680	-1600	2320	653	1020	325	51	0	0	0	0	390	430
18	5680	1600	-2320	-653	1020	325	52	0	0	0	0	390	430
19	2310	5510	-946	-2250	723	1550	53	0	0	0	0	404	334
20	-2310	-5510	946	2250	723	1550	54	0	0	0	0	404	334
21	0	0	0	0	314	347	55	0	0	0	0	420	264
22	0	0	0	0	314	347	56	0	0	0	0	420	264
23	0	0	0	0	326	270	57	-4640	-1380	1900	565	1120	386
24	0	0	0	0	326	270	58	4640	1380	-1900	-565	1120	386
25	0	0	0	0	338	213	59	2290	5300	-937	-2170	883	1820
26	0	0	0	0	338	213	60	-2290	-5300	937	2170	883	1820
27	-4070	-1210	1660	496	902	311	61	0	0	0	0	413	458
28	4070	1210	-1660	-496	902	311	62	0	0	0	0	413	458
29	1740	3990	-713	-1630	646	1310	63	0	0	0	0	391	326
30	-1740	-3990	713	1630	646	1310	64	0	0	0	0	391	326
31	0	0	0	0	340	375	65	0	0	0	0	368	233
32	0	0	0	0	340	375	66	0	0	0	0	368	233
33	0	0	0	0	352	291	67	-3030	-922	1240	377	804	287
34	0	0	0	0	352	291	68	3030	922	-1240	-377	804	287

Node No.	x		y		z		Node No.	x		y		z	
	L.C.1	L.C.2	L.C.1	L.C.2	L.C.1	L.C.2		L.C.1	L.C.2	L.C.1	L.C.2	L.C.1	L.C.2
69	3070	7160	-520	-1210	1040	2180	74	0	0	0	0	370	310
70	-3070	-7160	520	1210	1040	2180	75	0	0	0	0	304	194
71	0	0	0	0	433	484	76	0	0	0	0	304	194
72	0	0	0	0	433	484	77	-1370	-451	262	86	446	175
73	0	0	0	0	370	310	78	1370	451	-262	-86	446	175

Table 2. Material Properties

Note: Node (i) in Table 1 corresponds to Node (i + 2) in Figure 2.

Graphite Epoxy:

Density = 0.055 lbs/in^3

Layer Thickness = 0.005 in.

Allowable Stresses = 115,000 psi

Elastic Constants

E_{11} = 18.5 x 10^6 psi

E_{22} = 1.6 x 10^6 psi

G = 0.65 x 10^6 psi

ν = 0.25

Aluminum:

Density = 0.1 lbs/in^3

E = 10.5 x 10^6 psi

ν = 0.3

Table 1b. Wing Geometry

NODE	X	Y	Z
1	6.350000E+01	9.000000E+01	1.125000E+00
2	6.350000E+01	9.000000E+01	-1.125000E+00
3	7.083300E+01	9.000000E+01	1.313000E+00
4	7.083300E+01	9.000000E+01	-1.313000E+00
5	7.816700E+01	9.000000E+01	1.500000E+00
6	7.816700E+01	9.000000E+01	-1.500000E+00
7	8.550000E+01	9.000000E+01	1.313000E+00
8	8.550000E+01	9.000000E+01	-1.313000E+00
9	9.283300E+01	9.000000E+01	1.125000E+00
10	9.283300E+01	9.000000E+01	-1.125000E+00
11	6.350000E+01	9.000000E+01	1.125000E+00
12	6.350000E+01	9.000000E+01	-1.125000E+00
13	6.968600E+01	8.747100E+01	1.349000E+00
14	6.968600E+01	8.747100E+01	-1.349000E+00
15	7.609700E+01	8.485100E+01	1.586000E+00
16	7.609700E+01	8.485100E+01	-1.586000E+00
17	8.274600E+01	8.213300E+01	1.427000E+00
18	8.274600E+01	8.213300E+01	-1.427000E+00
19	8.964700E+01	7.931200E+01	1.259000E+00
20	8.964700E+01	7.931200E+01	-1.259000E+00
21	5.726600E+01	7.766900E+01	1.279000E+00
22	5.726600E+01	7.766900E+01	-1.279000E+00
23	6.399200E+01	7.492000E+01	1.532000E+00
24	6.399200E+01	7.492000E+01	-1.532000E+00
25	7.096200E+01	7.237100E+01	1.799000E+00
26	7.096200E+01	7.237100E+01	-1.799000E+00
27	7.819100E+01	6.911600E+01	1.617000E+00
28	7.819100E+01	6.911600E+01	-1.617000E+00
29	8.569200E+01	6.605900E+01	1.424000E+00
30	8.569200E+01	6.605900E+01	-1.424000E+00
31	5.103200E+01	6.537900E+01	1.433000E+00
32	5.103200E+01	6.537900E+01	-1.433000E+00
33	5.829700E+01	6.236900E+01	1.715000E+00
34	5.829700E+01	6.236900E+01	-1.715000E+00
35	6.582600E+01	5.929100E+01	2.012000E+00
36	6.582600E+01	5.929100E+01	-2.012000E+00
37	7.363500E+01	5.610000E+01	1.807000E+00
38	7.363500E+01	5.610000E+01	-1.807000E+00
39	8.173800E+01	5.278700E+01	1.590000E+00
40	8.173800E+01	5.278700E+01	-1.590000E+00
41	4.470000E+01	5.300800E+01	1.587000E+00
42	4.479900E+01	5.300800E+01	-1.587000E+00
43	5.260300E+01	4.981800E+01	1.898000E+00
44	5.260300E+01	4.981800E+01	-1.898000E+00
45	6.069100E+01	4.651200E+01	2.225000E+00
46	6.069100E+01	4.651200E+01	-2.225000E+00
47	6.907900E+01	4.308700E+01	1.997000E+00
48	6.907900E+01	4.308700E+01	-1.997000E+00
49	7.779400E+01	3.952500E+01	1.756000E+00
50	7.779400E+01	3.952500E+01	-1.756000E+00
51	8.655500E+01	4.067800E+01	1.742000E+00
52	8.655500E+01	4.067800E+01	-1.742000E+00
53	4.690800E+01	3.726700E+01	2.082000E+00
54	4.690800E+01	3.726700E+01	-2.082000E+00
55	5.555500E+01	3.373200E+01	2.438000E+00

Table 1b. Wing Geometry

NODE	X	Y	Z
56	5.555500E+01	3.373200E+01	-2.438000E+00
57	6.452300E+01	3.006700E+01	2.187000E+00
58	6.452300E+01	3.006700E+01	-2.187000E+00
59	7.383000E+01	2.626200E+01	1.922000E+00
60	7.383000E+01	2.626200E+01	-1.922000E+00
61	8.233100E+01	2.334700E+01	1.896000E+00
62	8.233100E+01	2.334700E+01	-1.896000E+00
63	9.121400E+01	2.471600E+01	2.265000E+00
64	9.121400E+01	2.471600E+01	-2.265000E+00
65	9.942000E+01	2.095300E+01	2.651000E+00
66	9.942000E+01	2.095300E+01	-2.651000E+00
67	9.996700E+01	1.705000E+01	2.376000E+00
68	9.996700E+01	1.705000E+01	-2.376000E+00
69	9.987600E+01	1.300000E+01	2.088000E+00
70	9.987600E+01	1.300000E+01	-2.088000E+00
71	9.516599E+01	1.417300E+01	2.073000E+00
72	9.516599E+01	1.417300E+01	-2.073000E+00
73	8.556298E+01	1.230400E+01	2.446000E+00
74	8.556298E+01	1.230400E+01	-2.446000E+00
75	7.618100E+01	1.040300E+01	2.827000E+00
76	7.618100E+01	1.040300E+01	-2.827000E+00
77	6.696399E+01	8.469000E+00	2.502000E+00
78	6.696399E+01	8.469000E+00	-2.502000E+00
79	6.793799E+01	6.500000E+00	2.169000E+00
80	6.793799E+01	6.500000E+00	-2.169000E+00
81	1.800000E+01	0.	2.250000E+00
82	1.800000E+01	0.	-2.250000E+00
83	3.000000E+01	0.	2.625000E+00
84	3.000000E+01	0.	-2.625000E+00
85	4.200000E+01	0.	3.000000E+00
86	4.200000E+01	0.	-3.000000E+00
87	5.400000E+01	0.	2.625000E+00
88	5.400000E+01	0.	-2.625000E+00
89	6.600000E+01	0.	2.250000E+00
90	6.600000E+01	0.	-2.250000E+00

REFERENCES

[1]V. B. Venkayya, N. S. Khot and V. S. Reddy, "Energy Distribution in an Optimum Structural Design," AFFDL-TR-68-156, 1969.

[2]V. B. Venkayya, "Design of Optimum Structures," J. Computers and Struct. 1, 265-309, 1971.

[3]V. B. Venkayya, N. S. Khot and L. Berke, "Application of Optimality Criteria Approaches to Automated Design of Large Practical Structures," Second Symp. Struct. Opt., AGARD-CP-123, Milan, Italy, 1973.

[4]N. S. Khot, V. B. Venkayya and L. Berke, "Optimum Design of Composite Structures with Stress and Deflection Constraints," AIAA Paper No. 75-141, Represented at AIAA 13th Aerospace Sciences Meeting, Pasadena, California, 1975.

[5]Khot, N. S., "Computer Program (OPTCOMP) for Optimization of Composite Structures for Minimum Weight Design," AFFDL-TR-76-149, February 1977.

[6]Venkayya, V. B., "Structural Optimization: A Review and Some Recommendations," Int. J. Numerical Methods in Engineering, Vol. 113, No. 2, pp 203-227, 1978.

[7]Gallagher, R. H., "Finite Element Analysis Fundamentals," Prentice-Hall Inc., Englewood Cliffs, N.J., 1975.

[8]Przemieniecki, J. S., "Theory of Matrix Structural Analysis," McGraw-Hill New York, 1968.

[9]Zienkiewicz, O. C., "The Finite Element Method in Engineering Science," McGraw-Hill Co., London, 1971.

[10]MacNeal, R. H., (Editor), "The NASTRAN Theoretical Manual, Levels 16 and 17," March 1976.

[11]Garvey, S. J., "The Quadrilateral Shear Panel," Aircraft Engineering, May 1951.

[12]Argyris, J. H., "Energy Theorems and Structural Analysis," Aircraft Engr., Vol. 26, pp. 347-356, 383-387, 394 (1954); Vol. 27, pp. 42-58, 80-94, 125-134, 145-158 (1955).

[13]Irons, B. M., "Engineering Application of Numerical Integration in Stiffness Method," AIAA Journal, Vol. 4, pp. 2035-2037, 1966.

[14]Bathe, K. J. and Wilson, E. L., "Numerical Methods in Finite Element Analysis," Prentice-Hall, Inc., Englewood Cliffs, N.J., 1976.

[15]Hohn, F. E., "Elementary Matrix Algebra," The McMillan Company, New York, 1958.

[16]Sandhu, R. S., "A Survey of Failure Theories of Isotropic and Anisotropic Materials," AFFDL-TR-72-71, pp. 19-22, September 1972.

[17]Tsai, S. W., "Strength Characteristics of Composite Materials," NASA CR-224, pp. 5-8, April 1965.

[18]Isakson, G. and Pardo, H., "ASOP-3: A Program for the Minimum Weight Design of Structures Subjected to Strength and Deflection Constraints," AFFDL-TR-76-157, December 1976.

O P T S T A T

PROGRAM LISTING

```
*DECK OPTSTAT
C     THE FOLLOWING DIM ARE FOR INTERNAL USE                                  OPTSTAT
      DIMENSION AA(3,3),EE(3,3),EK(12,12),EKK(12,12),B(12,12),C(12,12),       OPTSTAT
     1          XI(5),ETA(5),MAA(4),MBB(4),MCC(4),TRANG(4),IM(6),JM(6),       OPTSTAT
     2          TFR(6),DEFMAX(6),EEK(8,8),ALS(5),TFFR(4)                      OPTSTAT
C                                                                             OPTSTAT
C     THE FOLLOWING DIM PERTAIN TO THE NUMBER OF ELEMENTS (MEMBS)             OPTSTAT
      DIMENSION MA(260),MB(260),MC(260),MD(260),NNODES(260),AE(260),          OPTSTAT
     1AAE(260),ELENTH(260),STRENG(260)                                        OPTSTAT
      DIMENSION LAM(260),AEX(260),AEY(260),MYOUNG(260),XANG(260),             OPTSTAT
     1YANG(260),ZANG(260),ENGX(260),ENGY(260),ENGXY(260),AAEX(260),           OPTSTAT
     2AAEY(260),AEMNM(260),AEMAX(260),AEXMIN(260),AEYMIN(260),                OPTSTAT
     3AEXYMIN(260),NZDEG(260),NNDEG(260),NFDEG(260),LFLAG1(260),              OPTSTAT
     4LFLAG2(260),NKIND(260),STRN0(260,6),STRN90(260,6),
     5STRN45P(260,6),STRN45N(260,6)
C                                                                             OPTSTAT
C     THE FOLLOWING DIM PERTAIN TO THE NUMBER OF JOINTS                       OPTSTAT
      DIMENSION X(300),Y(300),Z(300)                                            OPTSTAT
C                                                                             OPTSTAT
C     THE FOLLOWING DIM PERTAIN TO THE NUMBER OF DEG OF FREEDOM (NN)          OPTSTAT
      DIMENSION DEFLMT(900),ICOL(900),IDIAG(900),ICOLS(900),IDIAGS(900),      OPTSTAT
     1   SK(20000)                                                             OPTSTAT
C                                                                             OPTSTAT
C     THE FOLLOWING DIM PERTAIN TO THE NUMBER OF BOUND. COND. (NBNDRY)        OPTSTAT
      DIMENSION IBND(50)                                                      OPTSTAT
C                                                                             OPTSTAT
C     THE FOLLOWING DIM PERTAIN TO THE NUMBER OF LOADING CONDITIONS (L)       OPTSTAT
      DIMENSION NJLODS(6),ELEENG(6),ENGSTR(6),KTR(6),EDR(12,6),               OPTSTAT
     1          EDDR(12,6),SX(6),SY(6),SXY(6),SSX(4,6),SSY(4,6),              OPTSTAT
     2          SSXY(4,6),EFSTRS(6),EFFSTR(4,6),EXM(6),SNMAX(6)
      DIMENSION ESRTIO(6),ELENG(6),ENGTOT(6),S(12,6)                          OPTSTAT
C                                                                             OPTSTAT
C     IF THE NUMBER OF LOADING CONDITIONS EXCEED 10, THEN CHANGE THE          OPTSTAT
C     DIMENSION OF TDR1,TDR2 IN SUBROUTINE RESTOR,ENGG IN SUBROUTINE          OPTSTAT
C      QLSTRS AND EX,EY,EXY IN SUBROUTINE STRESS                              OPTSTAT
      DIMENSION YOUNGM(20),POISON(20),RHO1(20),ELCNST(50),ALSTRS(100)         OPTSTAT
      DIMENSION NUFR(20),UDR(200,2), ENGST1(12,6),EDR1(12,12),ELENG1(20)      OPTSTAT
      DIMENSION TITLE(8)                                                      OPTSTAT
C     THE FOLLOWING DIM ARE FR(NN,L),DR(NN,L),DELTAR(NN,L),PDELR(NN,L)        OPTSTAT
      DIMENSION FR(900,6),DR(900,6)                                           OPTSTAT
C                                                                             OPTSTAT
C                                                                             OPTSTAT
C     THE FOLLOWING DIM IS MDEFEQ(NACTIVE,L)                                  OPTSTAT
      DIMENSION MDEFEQ(12,6),KDEFEQ(12)                                       OPTSTAT
C                                                                             OPTSTAT
      EQUIVALENCE (SK(1),NZDEG(1)),(SK(501),NNDEG(1)),                        OPTSTAT
     1(SK(1001),NFDEG(1)),(SK(1501),LFLAG1(1)),                               OPTSTAT
     2(SK(2001),LFLAG2(1)),(SK(2501),NKIND(1))                                OPTSTAT
C                                                                             OPTSTAT
C     NNMAX MUST BE THE DIMENSION OF FR,DR,              ICOL,IDIAG,          OPTSTAT
C     ICOLS,IDIAGS,DEFLMT                                                     OPTSTAT
      INTEGER TYPE                                                            OPTSTAT
      NNMAX = 900                                                             OPTSTAT
C     MAXSK MUST BE EQUAL TO OR GREATER THAN THE DIM OF SK                    OPTSTAT
      MAXSK=20000                                                              OPTSTAT
      NACTIVE = 12                                                            OPTSTAT
      READ(5,2)    NSTR                                                       OPTSTAT
      KSTR=1                                                                  OPTSTAT
    1 READ(5,6)TITLE                                                          OPTSTAT
      WRITE(6,6)TITLE                                                         OPTSTAT
    6 FORMAT(8A10)                                                            OPTSTAT
```

```
      READ(5,2)MEMBS,JOINTS,NBNDRY,LOADS,MM,LMTDSP,LMTCCL,INCHES,KIPS,    OPTSTAT
     1LSTCCL,NR,IAREAS,INSIST,LPRINT                                      OPTSTAT
      WRITE(6,2) MEMBS,JOINTS,NBNDRY,LOADS,MM,LMTDSP,LMTCCL,INCHES,       OPTSTAT
     1           KIPS,LSTCCL,NR,IAREAS,INSIST                             OPTSTAT
      READ(5,2)NMAT,NISOTR,INDANG,LAYERD,NCDPEL,NCDPND,INDMIN,KANLYZE     OPTSTAT
     1,MAXSZE,MNLAYR                                                      OPTSTAT
      WRITE(6,2)NMAT,NISOTR,INDANG,LAYERD,NCDPEL,NCDPND,INDMIN,KANLYZE    OPTSTAT
     1,MAXSZE,MNLAYR                                                      OPTSTAT
      READ(5,3)AEMNMM,DINCR,THKLAM,SPRDF                                  OPTSTAT
      IF(SPRDF .LT. 0.2)SPRDF=0.5                                         OPTSTAT
      WRITE(6,3)AEMNMM,DINCR,THKLAM ,SPRDF                                OPTSTAT
      ISOTRN=NMAT-NISOTR                                                  OPTSTAT
      READ(5,3) (YOUNGM(I),POISON(I),RHO1(I), I = 1,NMAT)                 OPTSTAT
      RAD=3.141592654/180.                                                OPTSTAT
      IF(NISOTR .EQ. 0)GO TO 7779                                         OPTSTAT
      KX=2*NISOTR                                                         OPTSTAT
      READ(5,3)(ELCNST(I),I=1,KX)                                         OPTSTAT
      IF(INDANG .EQ. 1)READ(5,3)(XANG(I),I=1,MEMBS)                       OPTSTAT
      IF(INDANG .EQ. 0)READ(5,3)(XANG(I),YANG(I),ZANG(I),I=1,MEMBS)       OPTSTAT
      IF(INDANG .EQ. 2)READ(5,3)AX,AY,AZ                                  OPTSTAT
      IF(INDANG .LE. 1)GO TO 7770                                         OPTSTAT
      DO 7777 I=1,MEMBS                                                   OPTSTAT
      XANG(I)=AX                                                          OPTSTAT
      YANG(I)=AY                                                          OPTSTAT
 7777 ZANG(I)=AZ                                                          OPTSTAT
 7770 CONTINUE                                                            OPTSTAT
      DO 7782 I = 1,MEMBS                                                 OPTSTAT
      IF(INDANG .EQ. 1)GO TO 7782                                         OPTSTAT
      YANG(I) = RAD          *YANG(I)                                     OPTSTAT
      ZANG(I)=RAD*ZANG(I)                                                 OPTSTAT
 7782 XANG(I) = RAD          *XANG(I)                                     OPTSTAT
 7779 CONTINUE                                                            OPTSTAT
      KX=5*NMAT                                                           OPTSTAT
      READ(5,3) (ALSTRS(I), I = 1,KX)                                     OPTSTAT
      DO 7781 I = 1,KX                                                    OPTSTAT
 7781 ALSTRS(I) = 1000.0*ALSTRS(I)                                        OPTSTAT
      IF(NCDPEL .EQ. 1)GO TO 7780                                         OPTSTAT
      READ(5,2)(NNODES(I),I=1,MEMBS)                                      OPTSTAT
      READ(5,2)(MA(I),I=1,MEMBS)                                          OPTSTAT
      READ(5,2)(MB(I),I=1,MEMBS)                                          OPTSTAT
      READ(5,2)(MC(I),I=1,MEMBS)                                          OPTSTAT
      READ(5,2)(MD(I),I=1,MEMBS)                                          OPTSTAT
      IF(NMAT .GT. 1)READ(5,2)(MYOUNG(I),I=1,MEMBS)                       OPTSTAT
      IF(LAYERD .GT. 0)READ(5,2)(LAM(I),I=1,MEMBS)                        OPTSTAT
      IF(IAREAS .EQ. 1)READ(5,3)(AE(I),I=1,MEMBS)                         OPTSTAT
      IF(IAREAS .EQ. 1 .AND. LAYERD .EQ. 1)READ(5,3)(AEX(I),I=1,MEMBS)    OPTSTAT
      IF(IAREAS .EQ. 1 .AND. LAYERD .EQ. 1)READ(5,3)(AEY(I),I=1,MEMBS)    OPTSTAT
      IF(INDMIN .EQ. 1)READ(5,3)(AEMNM(I),I=1,MEMBS)                      OPTSTAT
      GO TO 7785                                                          OPTSTAT
 7780 CONTINUE                                                            OPTSTAT
      DO 6000 I=1,MEMBS                                                   OPTSTAT
 6000 READ(5,7790)KX,NNODES(I),MYOUNG(I),MA(I),MB(I),MC(I),MD(I),LAM(I),  OPTSTAT
     1AE(I),AEX(I),AEY(I),AEMNM(I)                                        OPTSTAT
 7785 CONTINUE                                                            OPTSTAT
      IF(MAXSZE .EQ. 1)READ(5,3)(AEMAX(I),I=1,MEMBS)                      OPTSTAT
      IF(MNLAYR.EQ.1)READ(5,3)(AEXMIN(I),AEYMIN(I),AEXYMIN(I),I=1,MEMBS)  OPTSTAT
 7790 FORMAT(8I5,4F10.3)                                                  OPTSTAT
      WRITE(6,7791)                                                       OPTSTAT
 7791 FORMAT(1H1,///5H  MAT,8X,3HE11,9X,3HE22,8X,4HMU12,10X,2H G,9X,      OPTSTAT
     13HRHO,8X,4HTENS,8X,4HCOMP,8X,4HTENS,8X,4HCOMP,7X,5HSHEAR)           OPTSTAT
      IF(ISOTRN .EQ. 0)GO TO 7792                                         OPTSTAT
```

```
      DO 7793 I=1,ISOTRN                                                OPTSTAT
      KX=5*(I-1)+1                                                      OPTSTAT
      KY=KX+4                                                           OPTSTAT
 7793 WRITE(6,7794)I,YOUNGM(I),POISON(I),RHO1(I),(ALSTRS(J),J=KX,KY)    OPTSTAT
 7794 FORMAT(I4,E16.5,12X,E12.5,12X,6E12.5)                             OPTSTAT
 7792 IF(NISOTR .EQ. 0)GO TO 7795                                       OPTSTAT
      KH=ISOTRN+1                                                       OPTSTAT
      DO 7796 I=KH,NMAT                                                 OPTSTAT
      KX=5*(I-1)+1                                                      OPTSTAT
      KY=KX+4                                                           OPTSTAT
      KXX=2*(I-ISOTRN-1)+1                                              OPTSTAT
 7796 WRITE(6,7797)I,YOUNGM(I),ELCNST(KXX),POISON(I),ELCNST(KXX+1),     OPTSTAT
     1RHO1(I),(ALSTRS(J),J=KX,KY)                                       OPTSTAT
 7797 FORMAT(I4,E16.5,9E12.5)                                           OPTSTAT
 7795 CONTINUE                                                          OPTSTAT
      IF(AEMNMM .LE.0.00001)AEMNMM=0.01                                 OPTSTAT
      WRITE(6,7798)                                                     OPTSTAT
 7798 FORMAT(1H1,///5H MEMB,5H TYPE,5H  MAT,5H   MA,5H   MB,5H   MC,    OPTSTAT
     15H   MD,5H  LAM,10X,2HAE,9X,3HAEX,9X,3HAEY,5X,7HMINSIZE,8X,4HXANG, OPTSTAT
     28X,4HYANG,8X,4HZANG /)                                            OPTSTAT
      DO 7799 I=1,MEMBS                                                 OPTSTAT
      IF(LAYERD .EQ. 0)GO TO 7802                                       OPTSTAT
      IF(MNLAYR .EQ. 1)GO TO 7802                                       OPTSTAT
      IF(LAM(I) .NE. 1)GO TO 7802                                       OPTSTAT
      AEXMIN(I)=THKLAM                                                  OPTSTAT
      AEYMIN(I)=THKLAM                                                  OPTSTAT
      AEXYMIN(I)= 2.*THKLAM                                             OPTSTAT
 7802 CONTINUE                                                          OPTSTAT
      IF(IAREAS .NE. 1)AE(I)=1.                                         OPTSTAT
      IF(INDMIN .EQ. 0)AEMNM(I)=AEMNMM                                  OPTSTAT
      IF(NMAT .LE. 1)MYOUNG(I)=1                                        OPTSTAT
      IF(INDANG .NE. 1)GO TO 7801                                       OPTSTAT
      YANG(I)=0.                                                        OPTSTAT
      ZANG(I)=0.                                                        OPTSTAT
 7801 WRITE(6,7800)I,NNODES(I),MYOUNG(I),MA(I),MB(I),MC(I),MD(I),LAM(I), OPTSTAT
     1AE(I),AEX(I),AEY(I),AEMNM(I),XANG(I),YANG(I),ZANG(I)              OPTSTAT
 7800 FORMAT(I4,2I5,I6,4I5,E14.5,6E12.5)                                OPTSTAT
      AEMNM(I)=AEMNM(I)*10.**6                                          OPTSTAT
 7799 CONTINUE                                                          OPTSTAT
      IF(LAYERD .EQ. 0)GO TO 475                                        OPTSTAT
      IF(IAREAS .EQ. 1)GO TO 475                                        OPTSTAT
      DO 5464 I=1,MEMBS                                                 OPTSTAT
      IF(LAM(I) .EQ. 0)GO TO 5464                                       OPTSTAT
      AEX(I)=0.25                                                       OPTSTAT
      AEY(I)=0.25                                                       OPTSTAT
      IF(LAM(I) .LE. 1)GO TO 5464                                       OPTSTAT
      IF(LAM(I)-3)5465,5466,5467                                        OPTSTAT
 5465 AEX(I)=0.5                                                        OPTSTAT
      AEY(I)=0.5                                                        OPTSTAT
      GO TO 5464                                                        OPTSTAT
 5466 AEX(I)=0.                                                         OPTSTAT
      AEY(I)=0.                                                         OPTSTAT
      GO TO 5464                                                        OPTSTAT
 5467 AEX(I)=0.34                                                       OPTSTAT
      AEY(I)=0.                                                         OPTSTAT
      IF(LAM(I) .EQ. 4)GO TO 5464                                       OPTSTAT
      AEX(I)=0.                                                         OPTSTAT
      AEY(I)=0.34                                                       OPTSTAT
 5464 CONTINUE                                                          OPTSTAT
  475 CONTINUE                                                          OPTSTAT
      IF(NCDPND .EQ. 1)GO TO 7783                                       OPTSTAT
```

```
      IF(MM .LT. 3)GO TO 4                                          OPTSTAT
      READ(5,3)(X(I),Y(I),Z(I),I=1,JOINTS)                          OPTSTAT
      GO TO 7784                                                    OPTSTAT
    4 READ (5,3) (X(I),Y(I),I=1,JOINTS)                             OPTSTAT
      DO 11 I=1,JOINTS                                              OPTSTAT
   11 Z(I)=0.0                                                      OPTSTAT
      GO TO 7784                                                    OPTSTAT
 7783 DO 6001 I=1,JOINTS                                            OPTSTAT
 6001 READ(5,6002)KX,X(I),Y(I),Z(I)                                 OPTSTAT
 6002 FORMAT(I5,3F10.0)                                             OPTSTAT
 7784 CONTINUE                                                      OPTSTAT
      IF(INCHES .EQ. 1)GO TO 9                                      OPTSTAT
      DO 7 I=1,JOINTS                                               OPTSTAT
      X(I)=X(I)*12.0                                                OPTSTAT
    7 Y(I)=Y(I)*12.0                                                OPTSTAT
      IF (MM .EQ. 2) GO TO 9
      DO 70 I = 1,JOINTS
      Z(I) = Z(I)*12.0
   70 CONTINUE
    9 CONTINUE                                                      OPTSTAT
      DO 7786 I=1,JOINTS                                            OPTSTAT
 7786 WRITE(6,18)I,X(I),Y(I),Z(I)                                   OPTSTAT
   18 FORMAT(20X,I10,3F15.5)                                        OPTSTAT
      LAST=0                                                        OPTSTAT
      WTLAST=10000000.                                              OPTSTAT
      KCOUNT=1                                                      OPTSTAT
      NPAGE=0                                                       OPTSTAT
      LPCYCL=0                                                      OPTSTAT
      LDEFGN=1                                                      OPTSTAT
      NN=MM*JOINTS                                                  OPTSTAT
      NM=NN-NBNDRY                                                  OPTSTAT
      READ(5,2) (IBND(I),I=1,NBNDRY)                                OPTSTAT
      WRITE(6,5)                                                    OPTSTAT
      WRITE(6,1009)(IBND(I),I=1,NBNDRY)                             OPTSTAT
      DO 10 I=1,NN                                                  OPTSTAT
      DO 10 J=1,LOADS                                               OPTSTAT
      DR(I,J)=0                                                     OPTSTAT
   10 FR(I,J)=0                                                     OPTSTAT
      READ(5, 2) (NJLODS(I),I=1,LOADS)                              OPTSTAT
      DO 21 J=1,LOADS                                               OPTSTAT
      KH=NJLODS(J)                                                  OPTSTAT
   12 IF(KH-3)13,13.14                                              OPTSTAT
   13 KX=KH                                                         OPTSTAT
      GO TO 15                                                      OPTSTAT
   14 KX=3                                                          OPTSTAT
   15 READ(5,16) (TFR(I),IM(I),JM(I),I=1,KX)                        OPTSTAT
      DO 22 I=1,KX                                                  OPTSTAT
      KY=MM*JM(I)-MM+IM(I)                                          OPTSTAT
   22 FR(KY,J)=FR(KY,J)+TFR(I)                                      OPTSTAT
      KH=KH-KX                                                      OPTSTAT
      IF(KH)21,21,12                                                OPTSTAT
   21 CONTINUE                                                      OPTSTAT
      DO 50 I = 1,6                                                 OPTSTAT
      DO 50 J = 1,LOADS                                             OPTSTAT
   50 S(I,J) = 0.0                                                  OPTSTAT
      DO 51 I = 1,NN,MM                                             OPTSTA
      KX = I/MM + 1                                                 OPTSTA
      DO 51 J = 1,LOADS                                             OPTSTAT
      S(1,J) = S(1,J) + FR(I,J)                                     OPTSTAT
      S(2,J) = S(2,J) + FR(I+1,J)                                   OPTSTAT
      IF (MM .EQ. 2) GO TO 56
```

```
      S(3,J) = S(3,J) + FR(I+2,J)                                          OPTSTAT
      S(4,J) = S(4,J) - FR(I+1,J)*Z(KX) + FR(I+2,J)*Y(KX)                  OPTSTAT
      S(5,J) = S(5,J) + FR(I,J)*Z(KX) - FR(I+2,J)*X(KX)                    OPTSTAT
   56 S(6,J) = S(6,J) - FR(I,J)*Y(KX) + FR(I+1,J)*X(KX)                    OPTSTAT
   51 CONTINUE                                                             OPTSTAT
      WRITE(6,52)                                                          OPTSTAT
   52 FORMAT(///50X,24HSUMMARY OF APPLIED LOADS///)                        OPTSTAT
      WRITE(6,53)                                                          OPTSTAT
   53 FORMAT(20X,2HFX,15X,2HFY,15X,2HFZ,15X,2HMX,15X,2HMY,15X,             OPTSTAT
     12HMZ///)                                                             OPTSTAT
      DO 54 J = 1,LOADS                                                    OPTSTAT
      WRITE(6,55) (S(I,J), I = 1,6)                                        OPTSTAT
   54 CONTINUE                                                             OPTSTAT
   55 FORMAT(10X,6E17.7)                                                   OPTSTAT
      IF (LMTDSP - 1) 160,151,153                                          OPTSTAT
  151 READ(5,3) (DEFMAX(I),I=1,MM)                                         OPTSTAT
      DO 152 I=1,JOINTS                                                    OPTSTAT
      KX=MM*(I-1) +1                                                       OPTSTAT
      DO 152  J=1,MM                                                       OPTSTAT
      DEFLMT(KX)=DEFMAX(J)                                                 OPTSTAT
152   KX=KX+1                                                              OPTSTAT
      GO TO 160                                                            OPTSTAT
153   DO 154 I=1,NN                                                        OPTSTAT
154   DEFLMT(I)=1000.0                                                     OPTSTAT
      READ (5,   2 ) NLTDEF                                                OPTSTAT
      KH=NLTDEF                                                            OPTSTAT
155   IF(KH-3) 156,156 ,157                                                OPTSTAT
156   KX=KH                                                                OPTSTAT
      GO TO  158                                                           OPTSTAT
157   KX=3                                                                 OPTSTAT
158   READ(5,16 )(TFR(I),IM(I),JM(I),I=1,KX)                               OPTSTAT
      DO 159  I=1,KX                                                       OPTSTAT
      KY=MM*(JM(I)-1)+IM(I)                                                OPTSTAT
159   DEFLMT(KY)=TFR(I)                                                    OPTSTAT
      KH=KH-KX                                                             OPTSTAT
      IF(KH)160,160,155                                                    OPTSTAT
 160  CONTINUE                                                             OPTSTAT
      IF(KIPS .NE. 1)GO TO 666                                             OPTSTAT
      DO 17 I=1,NN                                                         OPTSTAT
      DO 17 J=1,LOADS                                                      OPTSTAT
  17  FR(I,J)=1000.0*FR(I,J)                                               OPTSTAT
 666  CONTINUE                                                             OPTSTAT
      STRAIN=50000./10.**3                                                 OPTSTAT
      DO 120 I=1,4                                                         OPTSTAT
      MAA(I)=I                                                             OPTSTAT
      MBB(I)=I+1                                                           OPTSTAT
 120  MCC(I)=5                                                             OPTSTAT
      MAA(4)=1                                                             OPTSTAT
      MBB(4)=4                                                             OPTSTAT
      CALL POP(MEMBS,JOINTS,MM,MA,MB,MC,MD,NNODES,ICOL,IDIAG,NONZRO,NR)    OPTSTAT
      IF(NONZRO .GT. MAXSK)GO TO 1000                                      OPTSTAT
      DO 24 I=1,NN                                                         OPTSTAT
      ICOLS(I)=ICOL(I)                                                     OPTSTAT
  24  IDIAGS(I)=IDIAG(I)                                                   OPTSTAT
      GO TO 422                                                            OPTSTAT
  19  DO 122 I=1,NN                                                        OPTSTAT
      ICOL(I)=ICOLS(I)                                                     OPTSTAT
 122  IDIAG(I)=IDIAGS(I)                                                   OPTSTAT
 422  ENGCAP=0                                                             OPTSTAT
      IF (KCOUNT .NE. 1) GO TO 424                                         OPTSTAT
      WRITE(6,427)                                                         OPTSTAT
```

```
  427 FORMAT(//5X,24HINITIAL  AREAS OF MEMBERS//)                          OPTSTAT
      WRITE(6,147) (AE(I),I = 1,MEMBS)                                    OPTSTAT
      GO TO 426                                                           OPTSTAT
  424 WRITE(6,148)                                                        OPTSTAT
      WRITE(6,147) (AE(I),I=1,MEMBS)                                      OPTSTAT
  426 DO 8 I=1,NONZRO                                                     OPTSTAT
    8 SK(I)=0                                                             OPTSTAT
      DO 400 L = 1,MEMBS                                                  OPTSTAT
   20 CALL COORD(MA(L),MB(L),MC(L),MD(L),X,Y,Z,AA,XI,ETA,AL,NNODES(L),0)  OPTSTAT
      CALL PREPAR(AE(L),AEX(L),AEY(L),ALS,ALSTRS,AX,AY,AZ,1.0,1.0,        OPTSTAT
     1EEK,E1,E2,SM,PMU,ELCNST,ESRTIO,ELENG,ISOTRN,NISOTR,KX,KY,LAM(L),    OPTSTAT
     2LAYERD,LOADS,MYOUNG(L),YOUNGM,POISON,NNODES(L),TFR,TFFR,0)          OPTSTAT
      IF(NNODES(L) .EQ. 2)GO TO 102                                       OPTSTAT
      DO 80 II=1,KX                                                       OPTSTAT
      CALL ELSTIC(E1,E2,PMU,SM,EE)                                        OPTSTAT
      TTHK=TFFR(II)                                                       OPTSTAT
      IF(TTHK .LE. 0.    )GO TO  80                                       OPTSTAT
      IF(NISOTR .EQ. 0)GO TO 26                                           OPTSTAT
      IF(MYOUNG(L) .LE. ISOTRN)GO TO 26                                   OPTSTAT
      CALL TRECON(EE,AA,XANG(L),YANG(L),ZANG(L),AX,AY,AZ,INDANG,II)       OPTSTAT
   26 IF(NNODES(L) .LT. 4)GO TO 27                                        OPTSTAT
      CALL QDRLTL(EK,EKK,TTHK ,ELENTH(L),MA(L),MB(L),MC(L),MD(L),MAA,     OPTSTAT
     1MBB,MCC,XI, ETA,NNODES(L),EE,TRANG,0)                               OPTSTAT
      GO TO 28                                                            OPTSTAT
   27 CALL PLSTIF(EK ,TTHK ,ELENTH(L),1,2,3             ,XI,ETA,EE,0.,0)  OPTSTAT
   28 IF(KX .LE. 1)GO TO 80                                               OPTSTAT
      DO 81 I=1,KY                                                        OPTSTAT
      DO 81 J=1,KY                                                        OPTSTAT
   81 EEK(I,J)=EEK(I,J)+EK(I,J)                                           OPTSTAT
   80 CONTINUE                                                            OPTSTAT
      IF(KX  .E. 1)GO TO 60                                               OPTSTAT
      DO 40 I=1,KY                                                        OPTSTAT
      DO 40 J=1,KY                                                        OPTSTAT
   40 EK(I,J)=EEK(I,J)                                                    OPTSTAT
   60 CONTINUE                                                            OPTSTAT
      CALL TRNSFM(EK,AA,B,C,MM,NNODES(L),12)                              OPTSTAT
      GO TO 103                                                           OPTSTAT
  102 CALL ELSTIF(AA,B,C,AE(L),MM,AL,E1)                                  OPTSTAT
      ELENTH(L)=AL                                                        OPTSTAT
  103 CALL ASEMBL(SK,C,MA(L),MB(L),MC(L),MD(L),MM,IDIAG,NNODES(L),12)     OPTSTAT
   30 FORMAT(/1X,9HBASEAE = ,6E15.5/)
      LX = MYOUNG(L)
      ENGLTA = AE(L)*(STRAIN**2)*RHO1(LX)*ELENTH(L)
      IF(NNODES(L) .GT. 4)ENGLTA=ENGLTA*SPRDF                             OPTSTAT
      ENGCAP=ENGCAP+ENGLTA                                                OPTSTAT
  400 CONTINUE                                                            OPTSTAT
C     CALL PRINTK(SK,IDIAG,NN)                                            OPTSTAT
      CALL BOUND2(SK,IBND,NN,NBNDRY,IDIAG,ICOL)                           OPTSTAT
      CALL REDUCE(FR,IBND,NN,NBNDRY,LOADS,NNMAX)                          OPTSTAT
      CALL  GAUSS(SK,FR,DR,ICOL,IDIAG,LOADS,NM,NNMAX,0)                   OPTSTAT
      DO  179 I=1,LOADS                                                   OPTSTAT
      ENGSTR(I)=0.                                                        OPTSTAT
      DO  179 J=1,NM                                                      OPTSTAT
  179 ENGSTR(I)=ENGSTR(I)+FR(J,I)*DR(J,I)                                 OPTSTAT
      IF(LOADS .EQ. 1) GO TO 173                                          OPTSTAT
      DO 172 I=2,LOADS                                                    OPTSTAT
      IF(ENGSTR(1).LT.ENGSTR(I)) ENGSTR(1) =ENGSTR(I)                     OPTSTAT
  172 CONTINUE                                                            OPTSTAT
  173 BASEAE=1000.0*SQRT(ENGSTR(1)/ENGCAP)                                OPTSTAT
      BASEA=BASEAE/10.**6                                                 OPTSTAT
      WRITE(6,30)BASEAE                                                   OPTSTAT
```

```
      CALL RESTOR(DR,IBND,NN,NBNDRY,LOADS,NNMAX)                        OPTSTAT
      CALL RESTOR(FR,IBND,NN,NBNDRY,LOADS,NNMAX)                        OPTSTAT
      IF(LMTDSP.EQ.0) GO TO 161                                         OPTSTAT
      DRATIO=0                                                          OPTSTAT
      DO 176 K=1,2                                                      OPTSTAT
      NDEFEQ=0                                                          OPTSTAT
      DO 176 I=1,NN                                                     OPTSTAT
      ADR=0.                                                            OPTSTAT
      DEFBAE=DEFLMT(I)*BASEAE                                           OPTSTAT
      DO 175  J=1,LOADS                                                 OPTSTAT
      IF(ABS(DR(I,J)).GT.ADR) ADR=ABS(DR(I,J))                          OPTSTAT
  175 CONTINUE                                                          OPTSTAT
      ADR=ADR/DEFBAE                                                    OPTSTAT
      IF(K-1) 191,191,192                                               OPTSTAT
  191 IF(DRATIO .LT. ADR) DRATIO=ADR                                    OPTSTAT
      GO TO 176                                                         OPTSTAT
  192 IF((DRATIO-ADR).GT.0.1  )GO TO 176                                OPTSTAT
      NDEFEQ=NDEFEQ+1                                                   OPTSTAT
      IF (NDEFEQ .GT. NACTIVE) STOP 777                                 OPTSTAT
  176 CONTINUE                                                          OPTSTAT
      BASEAE=BASEAE*DRATIO                                              OPTSTAT
      BASEA=BASEA*DRATIO                                                OPTSTAT
      LDEFGN=1                                                          OPTSTAT
  161 CONTINUE                                                          OPTSTAT
      WRITE(6,30)BASEAE                                                 OPTSTAT
      WEIGHT=0.                                                         OPTSTAT
      WMEMB = 0.0                                                       OPTSTAT
      WSHEAR = 0.0                                                      OPTSTAT
      WBAR = 0.0                                                        OPTSTAT
      RATINC=1.                                                         OPTSTAT
      MXMEMB = 0
      WRITE(6,36)                                                       OPTSTAT
   36 FORMAT(//,5X,9HMEMB. NO.,5X,15HSCALING FACTORS/)                  OPTSTAT
      DO 300 L=1,MEMBS                                                  OPTSTAT
      CALL COORD(MA(L),MB(L),MC(L),MD(L),X,Y,Z,AA,XI,ETA,AL,NNODES(L),0) OPTSTAT
      CALL ELFORC(AA,DR,EDR,MM,MA(L),MB(L),MC(L),MD(L),NNODES(L),LOADS,  OPTSTAT
     1NNMAX)                                                            OPTSTAT
      CALL PREPAR(AE(L),AEX(L),AEY(L),ALS,ALSTRS,AX,AY,AZ,1.0,BASEAE,    OPTSTAT
     1EEK,E1,E2,SM,PMU,ELCNST,ESRTIO,ELENG,ISOTRN,NISOTR,KX,KY,LAM(L),   OPTSTAT
     2LAYERD,LOADS,MYOUNG(L),YOUNGM,POISON,NNODES(L),TFR,TFFR,1)        OPTSTAT
      IF(NNODES(L) .EQ. 2)GO TO 213                                     OPTSTAT
      DO 180 II=1,KX                                                    OPTSTAT
      CALL ELSTIC(E1,E2,PMU,SM,EE)                                      OPTSTAT
      TTHK=TFFR(II)                                                     OPTSTAT
      IF(TTHK .LE. 0.    ) GO TO 180                                    OPTSTAT
      IF(NISOTR .EQ. 0)GO TO 126                                        OPTSTAT
      IF(MYOUNG(L) .LE. ISOTRN)GO TO 126                                OPTSTAT
      CALL TRECON(EE,AA,XANG(L),YANG(L),ZANG(L),AX,AY,AZ,INDANG,II)     OPTSTAT
  126 IF(NNODES(L) .LT. 4)GO TO 127                                     OPTSTAT
      CALL QDRLTL(EK,EKK,TTHK ,QUAD,MA(L),MB(L),MC(L),MD(L),MAA,MBB,MCC, OPTSTAT
     1XI, ETA,NNODES(L),EE,TRANG,1)                                     OPTSTAT
      CALL QLSTRS(EDR,EDDR,XI,ETA,MAA,MBB,MCC,SX,SY,SXY,EFSTRS,EXM,SNMAX
     1,EE,AX,AY,AZ,ALS,LOADS,SSX,SSY,SSXY,EFFSTR,KTR,EKK,ELEENG
     2,NNODES(L))
      DO 800 J=1,LOADS                                                  OPTSTAT
      EFSTRS(J)=0.                                                      OPTSTAT
      DO 801 I=1,4                                                      OPTSTAT
  801 EFSTRS(J)=EFSTRS(J)+TRANG(I)*EFFSTR(I,J)                          OPTSTAT
  800 EFSTRS(J)=EFSTRS(J)/QUAD                                          OPTSTAT
      GO TO 128                                                         OPTSTAT
  127 CALL STRESS(EDR,XI,ETA,1,2,3,SX,SY,SXY,EFSTRS EXM,EE,AX,AY,AZ,ALS,
```

```
     1LOADS,ELEENG,TRIANG,3)                                                    OPTSTAT
  128 IF(KX.LE.1) GO TO 180                                                     OPTSTAT
      ENGMAX=0.                                                                 OPTSTAT
      DO 802 J=1,LOADS                                                          OPTSTAT
      ESRTIO(J)=ESRTIO(J)+EFSTRS(J)*TTHK                                        OPTSTAT
      IF(ELEENG(J) .GT. ENGMAX)ENGMAX=ELEENG(J)                                 OPTSTAT
  802 ELENG(J)=ELENG(J)+ELEENG(J)*TTHK*0.5                                      OPTSTAT
      ENGMAX=ENGMAX*TTHK                                                        OPTSTAT
      IF(II .EQ. 1)ENGX(L)=ENGMAX                                               OPTSTAT
      IF(II .EQ. 2)ENGY(L)=ENGMAX                                               OPTSTAT
      IF(II .EQ. 3)ENGXY(L)=ENGMAX                                              OPTSTAT
      IF(II .EQ. 4)ENGXY(L)=ENGXY(L)+ENGMAX                                     OPTSTAT
  180 CONTINUE                                                                  OPTSTAT
      IF(KX.LE.1) GO TO 804                                                     OPTSTAT
      ENGMAX=0.                                                                 OPTSTAT
      DO 805  J=1,LOADS                                                         OPTSTAT
      IF(ELENG(J) .GT. ENGMAX)ENGMAX=ELENG(J)                                   OPTSTAT
      ESRTIO(J) =ESRTIO(J)/(AE(L)     )                                         OPTSTAT
  805 ENGTOT(J)=ENGTOT(J)+ELENG(J)                                              OPTSTAT
      GO TO 299                                                                 OPTSTAT
  804 CONTINUE                                                                  OPTSTAT
      ENGMAX=0.                                                                 OPTSTAT
      DO 807 J=1,LOADS                                                          OPTSTAT
      ESRTIO(J)=EFSTRS(J)                                                       OPTSTAT
      ELEENG(J)=ELEENG(J)*0.5*AE(L)                                             OPTSTAT
      IF(ELEENG(J) .GT. ENGMAX)ENGMAX=ELEENG(J)                                 OPTSTAT
  807 ENGTOT(J)=ENGTOT(J)+ELEENG(J)                                             OPTSTAT
      GO TO 299                                                                 OPTSTAT
  213 CONTINUE                                                                  OPTSTAT
      ENGMAX=0.                                                                 OPTSTAT
      DO 215 K=1,LOADS                                                          OPTSTAT
      SX(K)=     E1*(EDR(1,K)-EDR(2,K))/AL                                      OPTSTAT
      ESRTIO(K)=-SX(K)/ALS(1)                                                   OPTSTAT
      IF(SX(K) .GT. 0.)ESRTIO(K)=SX(K)/ALS(2)                                   OPTSTAT
      ELEENG(K)=(0.5*SX(K)**2/(        E1))*AL*AE(L)                            OPTSTAT
      IF(ELEENG(K) .GT. ENGMAX)ENGMAX=ELEENG(K)                                 OPTSTAT
  215 ENGTOT(K)=ENGTOT(K)+ELEENG(K)                                             OPTSTAT
  299 CONTINUE                                                                  OPTSTAT
      STRENG(L)=ENGMAX                                                          OPTSTAT
      AMAX=0.                                                                   OPTSTAT
      DO 298 J=1,LOADS                                                          OPTSTAT
      IF(ESRTIO(J) .GT. AMAX)AMAX=ESRTIO(J)                                     OPTSTAT
  298 CONTINUE                                                                  OPTSTAT
      IF(AMAX .GT. 1.)WRITE(6,34)L,AMAX                                         OPTSTAT
      IF(AMAX .LE. 1.)GO TO 297                                                 OPTSTAT
      BASEAE=BASEAE*AMAX                                                        OPTSTAT
      BASEA=BASEA*AMAX                                                          OPTSTAT
      MXMEMB=L                                                                  OPTSTAT
  297 CONTINUE                                                                  OPTSTAT
      KX=MYOUNG(L)                                                              OPTSTAT
      IF (NNODES(L) .EQ. 4 .OR. NNODES(L) .EQ. 3) WMEMB = WMEMB +               OPTSTAT
     1AE(L)*ELENTH(L)*RHO1(KX)                                                  OPTSTAT
      IF (NNODES(L) .EQ. 5) WSHEAR = WSHEAR + AE(L)*ELENTH(L)*RHO1(KX)          OPTSTAT
      IF (NNODES(L) .EQ. 2) WBAR = WBAR + AE(L)*ELENTH(L)*RHO1(KX)              OPTSTAT
      WEIGHT=WEIGHT+AE(L)*ELENTH(L)*RHO1(KX)                                    OPTSTAT
      RATINC=RATINC*AMAX                                                        OPTSTAT
  300 CONTINUE                                                                  OPTSTAT
      IF(RATINC .GT. DINCR)LDEFGN=0                                             OPTSTAT
   34 FORMAT(8X,I4,8X,E13.6)                                                    OPTSTAT
      IF(KANLYZE .EQ. 0)GO TO 404                                               OPTSTAT
      BAMAX=BASEA                                                               OPTSTAT
```

```
      WRITE(6,403)BAMAX                                                 OPTSTAT
  403 FORMAT(10X,33HMAXIMUM EFFECTIVE STRESS RATIO = ,E13.5)            OPTSTAT
      BASEA=1.                                                          OPTSTAT
      BASEAE=10.**6                                                     OPTSTAT
  404 CONTINUE                                                          OPTSTAT
      PERMBA=BASEA                                                      OPTSTAT
      IF(KANLYZE .EQ. 1)GO TO 402                                       OPTSTAT
      IF(MAXSZE .EQ. 0)GO TO 402                                        OPTSTAT
      DO 401 L=1,MEMBS                                                  OPTSTAT
      IF(AE(L)*BASEA .LE. AEMAX(L))GO TO 401                            OPTSTAT
      AX=AEMAX(L)/AE(L)                                                 OPTSTAT
      IF(AX  .GT. PERMBA)GO TO 401                                      OPTSTAT
      PERMBA=AX                                                         OPTSTAT
      MXMEMB=L                                                          OPTSTAT
  401 CONTINUE                                                          OPTSTAT
      AX=BASEA/PERMBA                                                   OPTSTAT
      WRITE(6,406)BASEA,PERMBA,AX,MXMEMB                                OPTSTAT
  406 FORMAT(5X,13HSCALE FACTORS, 5X,10HDESIRED = ,E11.4,5X,9HACTUAL = , OPTSTAT
     1E11.4,5X,8HRATIO = ,E11.4,5X,18HCRITICAL MEMBER = ,I10)           OPTSTAT
  402 CONTINUE                                                          OPTSTAT
      BASEA=PERMBA                                                      OPTSTAT
      BASEAE=BASEA*10.**6                                               OPTSTAT
      WMEMB = WMEMB*BASEA                                               OPTSTAT
      WSHEAR = WSHEAR*BASEA                                             OPTSTAT
      WBAR = WBAR*BASEA                                                 OPTSTAT
      WEIGHT=WEIGHT*BASEA                                               OPTSTAT
      WRITE(6,116) BASEAE,WEIGHT,WMEMB                                  OPTSTAT
      IF(LPCYCL .GE. 1)WRITE(6,149)LPCYCL                               OPTSTAT
      WRITE(6,143) KSTR,LOADS,KCOUNT,WSHEAR,WBAR                        OPTSTAT
      DO 224 I=1,NN                                                     OPTSTAT
      DO 224 J=1,LOADS                                                  OPTSTAT
  224 DR(I,J)=DR(I,J)/BASEAE                                            OPTSTAT
      IF(KANLYZE .EQ. 1)GO TO 250                                       OPTSTAT
      IF(LPCYCL .GT. LMTCCL)GO TO 250                                   OPTSTAT
      IF(KCOUNT .GT. LSTCCL .AND. LMTDSP .EQ. 0)GO TO 250               OPTSTAT
      IF(INSIST .EQ. 2  .AND.  LMTDSP .GT. 0)GO TO 119                  OPTSTAT
      IF(KCOUNT .LT. LSTCCL)GO TO 119                                   OPTSTAT
      IF(INSIST .EQ. 2)GO TO 119                                        OPTSTAT
C     IF(KCOUNT .EQ. 1) GO TO 113                                       OPTSTAT
      IF(INSIST .EQ. 1  .AND.  KCOUNT .LT. LSTCCL)GO TO 113             OPTSTAT
      IF(INSIST .EQ. 1  .AND.  KCOUNT .EQ. LSTCCL)GO TO 119             OPTSTAT
      IF(LPCYCL .GT. LMTCCL)GO TO 250                                   OPTSTAT
      IF(LPCYCL .GE. 1)GO TO 119                                        OPTSTAT
      IF(KCOUNT .GE. LSTCCL .AND. LMTDSP .EQ. 1)GO TO 119               OPTSTAT
      IF(LAST.GE.1) GO TO 112                                           OPTSTAT
      PCTWT= 0.0001*WEIGHT                                              OPTSTAT
      IF((WTLAST-WEIGHT) .GT. PCTWT .AND. LPCYCL .EQ. 0)GO TO113        OPTSTAT
      IF((WTLAST-WEIGHT) .LT. 0.    .AND. LPCYCL .EQ. 0)GO TO114        OPTSTAT
  112 IF(LMTDSP .EQ. 0 .OR. LDEFGN .EQ. 0)GO TO 250                     OPTSTAT
      LAST=0                                                            OPTSTAT
      GO TO 188                                                         OPTSTAT
  119 CONTINUE                                                          OPTSTAT
      KSAVE=0                                                           OPTSTAT
      IF(WTLAST .GT. WEIGHT)KSAVE=1                                     OPTSTAT
      IF(KCOUNT .LT. LSTCCL .AND. INSIST .LT. 2)GO TO 113               OPTSTAT
      IF(LPCYCL .LT. LMTCCL .AND. LMTDSP .GT. 0)GO TO 188               OPTSTAT
      IF(WEIGHT .LE. WTLAST)GO TO 250                                   OPTSTAT
  114 DO 182 I=1,MEMBS                                                  OPTSTAT
      IF(LAM(I) .GT. 0)AEX(I)=AAEX(I)                                   OPTSTAT
      IF(LAM(I) .GT. 0)AEY(I)=AAEY(I)                                   OPTSTAT
  182 AE(I)=AAE(I)                                                      OPTSTAT
```

```
      LAST=LAST+1                                                        OPTSTAT
      IF(LPCYCL .GE. 1)LPCYCL=LPCYCL+1                                   OPTSTAT
      GO TO 19                                                           OPTSTAT
  113 AMAXAE=0                                                           OPTSTAT
      DO 177 L=1,MEMBS                                                   OPTSTAT
      KX = MYOUNG(L)
      IF(KSAVE .EQ. 0)GO TO 183                                          OPTSTAT
      AAE(L)=AE(L)                                                       OPTSTAT
      IF(LAYERD .EQ. 0)GO TO 183                                         OPTSTAT
      AAEX(L)=AEX(L)                                                     OPTSTAT
      AAEY(L)=AEY(L)                                                     OPTSTAT
  183 CONTINUE                                                           OPTSTAT
      STRENG(L)=STRENG(L)/BASEAE                                         OPTSTAT
      ENGLTA=AE(L)*ELENTH(L)*(STRAIN**2)*RH01(KX)
      IF(NNODES(L) .GT. 4)ENGLTA=ENGLTA*SPRDF                            OPTSTAT
      AE(L)=1000.0*AE(L)*SQRT(STRENG(L)/ENGLTA    )                      OPTSTAT
  177 IF(AMAXAE.LT.AE(L))AMAXAE =AE(L)                                   OPTSTAT
      KH=0                                                               OPTSTAT
      DO 178 I=1,MEMBS                                                   OPTSTAT
      AE(I)=AE(I)/AMAXAE                                                 OPTSTAT
      IF((AE(I)*BASEAE) .GT. AEMNM(I))GO TO 178                          OPTSTAT
      KH=KH+1                                                            OPTSTAT
      AE(I)=AEMNM(I)/BASEAE                                              OPTSTAT
  178 CONTINUE                                                           OPTSTAT
      IF(KH .LT. MEMBS)GO TO 181                                         OPTSTAT
      DO 184 I=1,MEMBS                                                   OPTSTAT
      AEMNM(I)=AEMNM(I)/10.                                              OPTSTAT
  184 AE(I)=0.1                                                          OPTSTAT
  181 IF(MAXSZE .EQ. 0)GO TO 174                                         OPTSTAT
      DO 171 L=1,MEMBS                                                   OPTSTAT
      IF(AE(L)*BASEA .GT. AEMAX(L))AE(L)=AEMAX(L)/BASEA                  OPTSTAT
  171 CONTINUE                                                           OPTSTAT
  174 CONTINUE                                                           OPTSTAT
      IF(LAYERD .EQ. 0)GO TO 169                                         OPTSTAT
      CALL LMSIZE(AE,AEX,AEY,BASEAE,BASEA,ENGX,ENGY,ENGXY,ELENTH,        OPTSTAT
     1LAM,MEMBS,TFR,AEXMIN,AEYMIN,AEXYMIN,MNLAYR)                        OPTSTAT
  169 CONTINUE                                                           OPTSTAT
      IF(KSAVE .EQ. 1)WTLAST=WEIGHT                                      OPTSTAT
      KCOUNT=KCOUNT+1                                                    OPTSTAT
      GO TO 19                                                           OPTSTAT
  188 CONTINUE                                                           OPTSTAT
      IF(WEIGHT .LE. WTLAST)GO TO 189                                    OPTSTAT
      AMAXAE=0.                                                          OPTSTAT
      DO 190 L=1,MEMBS                                                   OPTSTAT
      AE(L)=(AE(L)+AAE(L))/2.                                            OPTSTAT
      IF(AE(L) .GT. AMAXAE)AMAXAE=AE(L)                                  OPTSTAT
      IF(LAM(L) .GT. 0)AEX(L)=(AEX(L)+AAEX(L))/2.                        OPTSTAT
      IF(LAM(L) .GT. 0)AEY(L)=(AEY(L)+AAEY(L))/2.                        OPTSTAT
  190 CONTINUE                                                           OPTSTAT
      DO 195 L=1,MEMBS                                                   OPTSTAT
      AE(L)=AE(L)/AMAXAE                                                 OPTSTAT
      IF(AE(L)*BASEAE .LT. AEMNM(L))AE(L)=AEMNM(L)/BASEAE                OPTSTAT
  195 CONTINUE                                                           OPTSTAT
      WRITE(6,170)                                                       OPTSTAT
  170 FORMAT(//5X,12HSTEP REDUCED//)                                     OPTSTAT
      KCOUNT=KCOUNT+1                                                    OPTSTAT
      LPCYCL=LPCYCL+1                                                    OPTSTAT
      GO TO 19                                                           OPTSTAT
  189 DO 226 I=1,NN                                                      OPTSTAT
      DO 226 J=1,LOADS                                                   OPTSTAT
  226 DR(I,J)=DR(I,J)*DINCR                                              OPTSTAT
```

```
      KX=0                                                                OPTSTAT
      DO 254 I=1,NN                                                       OPTSTAT
      DO 253 J=1,LOADS                                                    OPTSTAT
      IF(ABS(DR(I,J)) .LE. DEFLMT(I))GO TO 253                            OPTSTAT
      KX=KX+1                                                             OPTSTAT
      NUFR(KX)=I                                                          OPTSTAT
      GO TO 254                                                           OPTSTAT
  253 CONTINUE                                                            OPTSTAT
  254 CONTINUE                                                            OPTSTAT
      NDUMMY=KX                                                           OPTSTAT
      WRITE(6,1012)NDUMMY,(NUFR(I),I=1,NDUMMY)                            OPTSTAT
 1012 FORMAT(/,5X,9HDUMMY = ,I6,/,5X,4HNUFR/,(10I13))
      IF(NDUMMY .EQ. 0)GO TO 250                                          OPTSTAT
      IF(NDUMMY .GT. NACTIVE)GO TO 1000                                   OPTSTAT
      DO 252 J=1,NDUMMY                                                   OPTSTAT
      NX=NUFR(J)                                                          OPTSTAT
      DO 252 I=1,NN                                                       OPTSTAT
      UDR(I,J)=0.                                                         OPTSTAT
      IF(I .EQ. NX)UDR(I,J)=1.                                            OPTSTAT
  252 CONTINUE                                                            OPTSTAT
      CALL REDUCE(UDR   ,IBND,NN,NBNDRY,NDUMMY,NNMAX)                     OPTSTAT
      CALL GAUSS1(SK,     UDR,ICOL,IDIAG,NDUMMY,NM,NNMAX)                 OPTSTAT
      CALL RESTOR(UDR   ,IBND ,NN,NBNDRY,NDUMMY,NNMAX)                    OPTSTAT
      DO 258 I=1,NDUMMY                                                   OPTSTAT
      KX=NUFR(I)                                                          OPTSTAT
      J1=0                                                                OPTSTAT
      DO 257 J=1,LOADS                                                    OPTSTAT
      IF(ABS(DR(KX,J)) .LE. DEFLMT(KX))GO TO 257                          OPTSTAT
      J1=J1+1                                                             OPTSTAT
      ENGST1(I,J1)=DEFLMT(KX)                                             OPTSTAT
      KDEFEQ(I)=J1                                                        OPTSTAT
      IF(DR(KX,J))255,255,256                                             OPTSTAT
  255 MDEFEQ(I,J1)=-J                                                     OPTSTAT
      GO TO 257                                                           OPTSTAT
  256 MDEFEQ(I,J1)=J                                                      OPTSTAT
  257 CONTINUE                                                            OPTSTAT
  258 CONTINUE                                                            OPTSTAT
      DO 280 L=1,MEMBS                                                    OPTSTAT
      CALL COORD(MA(L),MB(L),MC(L),MD(L),X,Y,Z,AA,XI,ETA,AL,NNODES(L),0)  OPTSTAT
      CALL ELFORC(AA,DR,EDR,MM,MA(L),MB(L),MC(L),MD(L),NNODES(L),LOADS,    OPTSTAT
     1NNMAX)                                                              OPTSTAT
      CALL ELFORC(AA,UDR,EDR1,MM,MA(L),MB(L),MC(L),MD(L),NNODES(L),       OPTSTAT
     1NDUMMY,NNMAX)                                                       OPTSTAT
```

```
      CALL PREPAR(AE(L),AEX(L),AEY(L),ALS,ALSTRS,AX,AY,AZ,1.0,1.0,        OPTSTAT
     1EEK,E1,E2,SM,PMU,ELCNST,ESRTIO,ELENG,ISOTRN,NISOTR,KX,KY,LAM(L),    OPTSTAT
     2LAYERD,LOADS,MYOUNG(L),YOUNGM,POISON,NNODES(L),TFR,TFFR,0)          OPTSTAT
      IF(NNODES(L) .EQ. 2)GO TO 1102                                      OPTSTAT
      DO 1180 II=1,KX                                                     OPTSTAT
      CALL ELSTIC(E1,E2,PMU,SM,EE)                                        OPTSTAT
      TTHK=TFFR(II)                                                       OPTSTAT
      IF(TTHK .LE. 0.    )GO TO  1180                                     OPTSTAT
      IF(NISOTR .EQ. 0)GO TO 1126                                         OPTSTAT
      IF(MYOUNG(L) .LE. ISOTRN)GO TO 1126                                 OPTSTAT
      CALL TRECON(EE,AA,XANG(L),YANG(L),ZANG(L),AX,AY,AZ,INDANG,II)       OPTSTAT
1126  IF(NNODES(L) .LT. 4)GO TO 1127                                      OPTSTAT
      CALL QDRLTL(EK,EKK,TTHK ,QUAD,MA(L),MB(L),MC(L),MD(L),MAA,MBB,MCC,  OPTSTAT
     1XI, ETA,NNODES(L),EE,TRANG,2)                                       OPTSTAT
      GO TO 1128                                                          OPTSTAT
1127  CALL PLSTIF(EK ,TTHK ,TRIANG, 1,2,3               ,XI,ETA,EE,0.,0)  OPTSTAT
1128  IF(KX .LE. 1)GO TO 1180                                             OPTSTAT
      CALL UNITEG(EK,ENGX(L),ENGY(L),ENGXY(L),EDR,EDR1,ELENG1,ENGST1,     OPTSTAT
     1LOADS,KDEFEQ,MDEFEQ,NDUMMY,NACTIVE,NNODES(L), S,STRENG(L),II,1)     OPTSTAT
      DO 1181 I=1,KY                                                      OPTSTAT
      DO 1181 J=1,KY                                                      OPTSTAT
1181  EEK(I,J)=EEK(I,J)+EK(I,J)                                           OPTSTAT
1180  CONTINUE                                                            OPTSTAT
      IF(KX .LE. 1)GO TO 1103                                             OPTSTAT
      DO 1140 I=1,KY                                                      OPTSTAT
      DO 1140 J=1,KY                                                      OPTSTAT
1140  EK(I,J)=EEK(I,J)                                                    OPTSTAT
1160  CONTINUE                                                            OPTSTAT
      GO TO 1103                                                          OPTSTAT
1102  DO 1105 K=1,LOADS                                                   OPTSTAT
      S(1,K)=AE(L)* E1*      (EDR(1,K)-EDR(2,K))/AL                       OPTSTAT
      S(2,K)=-S(1,K)                                                      OPTSTAT
1105  CONTINUE                                                            OPTSTAT
1103  CONTINUE                                                            OPTSTAT
      CALL UNITEG(EK,ENGX(L),ENGY(L),ENGXY(L),EDR,EDR1,ELENG1,ENGST1,     OPTSTAT
     1LOADS,KDEFEQ,MDEFEQ,NDUMMY,NACTIVE,NNODES(L), S,STRENG(L),II,0)     OPTSTAT
 280  CONTINUE                                                            OPTSTAT
 277  AMAXAE=0.                                                           OPTSTAT
      DO 275 L=1,MEMBS                                                    OPTSTAT
      KX = MYOUNG(L)
      ENGLTA=AE(L)*ELENTH(L)*(STRAIN**2)*RH01(KX)
      IF(NNODES(L) .GT. 4)ENGLTA=ENGLTA*SPRDF                             OPTSTAT
      IF(KSAVE .EQ. 0)GO TO 374                                           OPTSTAT
      AAE(L)=AE(L)                                                        OPTSTAT
      IF(LAYERD .EQ. 0)GO TO 374                                          OPTSTAT
      AAEX(L)=AEX(L)                                                      OPTSTAT
      AAEY(L)=AEY(L)                                                      OPTSTAT
 374  CONTINUE                                                            OPTSTAT
      IF(STRENG(L) .GT. 0.)GO TO 278                                      OPTSTAT
      AE(L)=0.                                                            OPTSTAT
      GO TO 275                                                           OPTSTAT
 278  AE(L)=1000.*AE(L)*SQRT(STRENG(L)/ENGLTA    )                       OPTSTAT
 275  IF(AMAXAE .LT. AE(L))AMAXAE=AE(L)                                   OPTSTAT
      DO 276 L=1,MEMBS                                                    OPTSTAT
      AE(L)=AE(L)/AMAXAE                                                  OPTSTAT
      IF(AE(L)*BASEAE  .GT. AEMNM(L))GO TO 276                            OPTSTAT
      AE(L)=AEMNM(L)/BASEAE                                               OPTSTAT
 276  CONTINUE                                                            OPTSTAT
      IF(LAYERD .EQ. 0)GO TO 139                                          OPTSTAT
      CALL LMSIZE(AE,AEX,AEY,1.,     BASEA,ENGX,ENGY,ENGXY,ELENTH,        OPTSTAT
     1LAM,MEMBS,TFR,AEXMIN,AEYMIN,AEXYMIN,MNLAYR)                         OPTSTAT
```

```
139  CONTINUE                                                          OPTSTAT
     IF(KSAVE .EQ. 1)WTLAST=WEIGHT                                     OPTSTAT
     KCOUNT=KCOUNT+1                                                   OPTSTAT
     LPCYCL=LPCYCL+1                                                   OPTSTAT
     GO TO 19                                                          OPTSTAT
250  CONTINUE                                                          OPTSTAT
     NPAGE=1                                                           OPTSTAT
     LINES = 1                                                         OPTSTAT
     DO 1501 I=1,LOADS                                                 OPTSTAT
1501 ENGTOT(I)=0.                                                      OPTSTAT
     DO 600 L=1,MEMBS                                                  OPTSTAT
     CALL COORD(MA(L),MB(L),MC(L),MD(L),X,Y,Z,AA,XI,ETA,AL,NNODES(L),0) OPTSTAT
     CALL ELFORC(AA,DR,EDR,MM,MA(L),MB(L),MC(L),MD(L),NNODES(L),LOADS,  OPTSTAT
    1NNMAX)                                                            OPTSTAT
     AAE(L)=AE(L)*BASEA                                                OPTSTAT
     TYPE = NNODES(L)*10 + MYOUNG(L)                                   OPTSTAT
     IF(MYOUNG(L) .GE. 10)TYPE=NNODES(L)*100+MYOUNG(L)                 OPTSTAT
  86 IF((LINES+LOADS) .LT. 54 .AND. L .GT. 1)GO TO 84                  OPTSTAT
     LINES=1                                                           OPTSTAT
     WRITE(6,98)NPAGE                                                  OPTSTAT
     NPAGE=NPAGE+1                                                     OPTSTAT
     WRITE(6,83)                                                       OPTSTAT
     WRITE(6,85)                                                       OPTSTAT
  84 CONTINUE                                                          OPTSTAT
     CALL PREPAR(AE(L),AEX(L),AEY(L),ALS,ALSTRS,AX,AY,AZ,BASEA,1.0,    OPTSTAT
    1EEK,E1,E2,SM,PMU,ELCNST,ESRTIO,ELENG,ISOTRN,NISOTR,KX,KY,LAM(L),  OPTSTAT
    2LAYERD,LOADS,MYOUNG(L),YOUNGM,POISON,NNODES(L),TFR,TFFR,1)        OPTSTAT
     IF(NNODES(L) .EQ. 2)GO TO 513                                     OPTSTAT
     DO 580 II=1,KX                                                    OPTSTAT
     CALL ELSTIC(E1,E2,PMU,SM,EE)                                      OPTSTAT
     TTHK=TFFR(II)                                                     OPTSTAT
     IF(TTHK .LE. 0.     )GO TO 580                                    OPTSTAT
     IF(NISOTR .EQ. 0)GO TO 526                                        OPTSTAT
     IF(MYOUNG(L) .LE. ISOTRN)GO TO 526                                OPTSTAT
     CALL TRECON(EE,AA,XANG(L),YANG(L),ZANG(L),AX,AY,AZ,INDANG,II)     OPTSTAT
526  IF(NNODES(L) .LT. 4)GO TO 527                                     OPTSTAT
     CALL QDRLTL(EK,EKK,TTHK ,QUAD,MA(L),MB(L),MC(L),MD(L),MAA,MBB,MCC, OPTSTAT
    1XI, ETA,NNODES(L),EE,TRANG,1)                                     OPTSTAT
     CALL QLSTRS(EDR,EDDR,XI,ETA,MAA,MBB,MCC,SX,SY,SXY,EFSTRS,EXM,SNMAX
    1,EE,AX,AY,AZ,ALS,LOADS,SSX,SSY,SSXY,EFFSTR,KTR,EKK,ELEENG,
    2NNODES(L))
     DO 700 J=1,LOADS                                                  OPTSTAT
     IF (II .EQ. 1) STRNO(L,J) = SNMAX(J)
     IF (II .EQ. 2) STRN90(L,J) = SNMAX(J)
     IF (II .EQ. 3) STRN45P(L,J) = SNMAX(J)
     IF (II .EQ. 4) STRN45N(L,J) = SNMAX(J)
     EFSTRS(J)=0.                                                      OPTSTAT
     DO 501 I=1,4                                                      OPTSTAT
501  EFSTRS(J)=EFSTRS(J)+TRANG(I)*EFFSTR(I,J)                          OPTSTAT
700  EFSTRS(J)=EFSTRS(J)/QUAD                                          OPTSTAT
     GO TO 528                                                         OPTSTAT
527  CALL STRESS(EDR,XI,ETA,1,2,3,SX,SY,SXY,EFSTRS,EXM,EE,AX,AY,AZ,ALS,
    1LOADS,ELEENG,TRIANG,3)                                            OPTSTAT
528  IF(KX.LE.1) GO TO 580                                             OPTSTAT
     DO 502 J=1,LOADS                                                  OPTSTAT
     IF (II .EQ. 1) STRNO(L,J) = EXM(J)
     IF (II .EQ. 2) STRN90(L,J) = EXM(J)
     IF (II .EQ. 3) STRN45P(L,J) = EXM(J)
     IF (II .EQ. 4) STRN45N(L,J) = EXM(J)
     ESRTIO(J)=ESRTIO(J)+EFSTRS(J)*TTHK                                OPTSTAT
502  ELENG(J)=ELENG(J)+ELEENG(J)*TTHK*0.5                              OPTSTAT
```

```
 580 CONTINUE                                                          OPTSTAT
     IF(KX.LE.1) GO TO 504                                             OPTSTAT
     ADR = (AE(L)*BASEA)/THKLAM                                        OPTSTAT
     LAM(L) = ADR                                                      OPTSTAT
     BDR = LAM(L)                                                      OPTSTAT
     IF ((ADR - BDR) .GT. 0.2) LAM(L) = LAM(L) + 1                     OPTSTAT
     DO 505  J=1,LOADS                                                 OPTSTAT
     ESRTIO(J) =ESRTIO(J)/(AE(L)*BASEA     )                           OPTSTAT
 505 ENGTOT(J)=ENGTOT(J)+ELENG(J)                                      OPTSTAT
     GO TO 506                                                         OPTSTAT
 504 DO 507 J=1,LOADS                                                  OPTSTAT
     ELEENG(J)=ELEENG(J)*0.5*AE(L)*BASEA                               OPTSTAT
 507 ENGTOT(J)=ENGTOT(J)+ELEENG(J)                                     OPTSTAT
 506 IF(NNODES(L).LT.4) GO TO 508                                      OPTSTAT
     IF(KX.LE.1) GO TO 509                                             OPTSTAT
     WRITE(6,187) L,AAE(L),QUAD,TYPE,MA(L),MB(L),MC(L),MD(L),LAM(L),   OPTSTAT
    1TFFR(1),AEX(L),TFFR(2),AEY(L),ESRTIO(1),(TFR(I),I=1,5),ELENG(1)   OPTSTAT
     IF (KANLYZE .EQ. 1) GO TO 46                                      OPTSTAT
     CALL LAYCALC(L,AAE(L),LAM(L),TFFR(1),TFFR(2),NZDEG,NNDEG,NFDEG,   OPTSTAT
    1THKLAM,LFLAG1,LFLAG2,NKIND,NCOUNT)                                OPTSTAT
  46 IF(LOADS.EQ.1) GO TO 600                                          OPTSTAT
     DO 311 K=2,LOADS                                                  OPTSTAT
 311 WRITE(6,194) ESRTIO(K),ELENG(K)                                   OPTSTAT
     GO TO 600                                                         OPTSTAT
 509 CONTINUE                                                          OPTSTAT
     KX=KTR(1)                                                         OPTSTAT
     IF (NNODES(L) .EQ. 5) GO TO 650                                   OPTSTAT
     WRITE(6,87) L,AAE(L),QUAD,TYPE,MA(L),MB(L),MC(L),MD(L),SSX(KX,1),  OPTSTAT
    1SSY(KX,1),SSXY(KX,1),EFSTRS(1),(TFR(I),I=1,5),ELEENG(1)           OPTSTAT
     GO TO 655                                                         OPTSTAT
 650 WRITE(6,82) L,AAE(L),QUAD,TYPE,MA(L),MB(L),MC(L),MD(L),           OPTSTAT
    1SSXY(KX,1),EFSTRS(1),(TFR(I),I=1,5),ELEENG(1)                     OPTSTAT
 655 IF(LOADS .EQ. 1)GO TO 600                                         OPTSTAT
     DO 211 K=2,LOADS                                                  OPTSTAT
     KX=KTR(K)                                                         OPTSTAT
     IF (NNODES(L) .EQ. 5) GO TO 657                                   OPTSTAT
     WRITE(6,94)SSX(KX,K),SSY(KX,K),SSXY(KX,K),EFSTRS(K),ELEENG(K)     OPTSTAT
     GO TO 211                                                         OPTSTAT
 657 WRITE(6,90) SSXY(KX,K),EFSTRS(K),ELEENG(K)                        OPTSTAT
 211 CONTINUE                                                          OPTSTAT
     GO TO 600                                                         OPTSTAT
 508 CONTINUE                                                          OPTSTAT
     IF(KX.LE.1) GO TO 510                                             OPTSTAT
     WRITE(6,186)L,AAE(L),TRIANG,TYPE,MA(L),MB(L),MC(L),LAM(L),TFFR(1),  OPTSTAT
    1AEX(L),TFFR(2),AEY(L),ESRTIO(1),(TFR(I),I=1,5),ELENG(1)           OPTSTAT
     IF (KANLYZE .EQ. 1) GO TO 48                                      OPTSTAT
     CALL LAYCALC(L,AAE(L),LAM(L),TFFR(1),TFFR(2),NZDEG,NNDEG,NFDEG,   OPTSTAT
    1THKLAM,LFLAG1,LFLAG2,NKIND,NCOUNT)                                OPTSTAT
  48 IF(LOADS.EQ.1) GO TO 600                                          OPTSTAT
     DO 312 K=2,LOADS                                                  OPTSTAT
 312 WRITE(6,194) ESRTIO(K),ELENG(K)                                   OPTSTAT
     GO TO 600                                                         OPTSTAT
 510 CONTINUE                                                          OPTSTAT
     WRITE(6,88) L,AAE(L),TRIANG,TYPE,MA(L),MB(L),MC(L),SX(1),SY(1),   OPTSTAT
    1SXY(1),EFSTRS(1),(TFR(I),I=1,5),ELEENG(1)                         OPTSTAT
     IF(LOADS .EQ. 1)GO TO 600                                         OPTSTAT
     DO 212 K=2,LOADS                                                  OPTSTAT
 212 WRITE(6,94)SX(K),SY(K),SXY(K),EFSTRS(K),ELEENG(K)                 OPTSTAT
     GO TO 600                                                         OPTSTAT
 513 DO 515 K=1,LOADS                                                  OPTSTAT
     SX(K)=    E1*(EDR(1,K)-EDR(2,K))/AL                               OPTSTAT
```

```
      EFSTRS(K)=-SX(K)/ALS(1)                                                    OPTSTAT
      IF(SX(K) .GT. 0.)EFSTRS(K)=SX(K)/ALS(2)                                    OPTSTAT
      ELEENG(K)=(0.5*SX(K)**2/E1 )*AL*AE(L)*BASEA                                OPTSTAT
  515 ENGTOT(K)=ENGTOT(K)+ELEENG(K)                                              OPTSTAT
      WRITE(6,89) L,AAE(L),AL,TYPE,MA(L),MB(L),SX(1),EFSTRS(1),TFR(1),           OPTSTAT
     1TFR(2),ELEENG(1)                                                           OPTSTAT
      IF(LOADS .EQ. 1)GO TO 600                                                  OPTSTAT
      DO 214 K=2,LOADS                                                           OPTSTAT
  214 WRITE(6,93)SX(K),EFSTRS(K),ELEENG(K)                                       OPTSTAT
  600 LINES=LINES+LOADS+1                                                        OPTSTAT
  883 FORMAT(1X,4HMEMB,2X,5HTHICK,3X,4HAREA,2X,4HTYPE,1X,2HMA,2X,
     1       2HMB,2X,2HMC,2X,2HMD,2X,9HSTRAIN(0),2X,10HSTRAIN(90),2X,
     2       11HSTRAIN(+45),2X,11HSTRAIN(-45))
      LINES = 1
      DO 6600 L = 1,MEMBS
      TYPE = NNODES(L)*10 + MYOUNG(L)
      IF ((LINES + LOADS) .LT. 54 .AND. L .GT. 1) GO TO 884
      LINES = 1
      NPAGE = NPAGE + 1
      WRITE(6,98)NPAGE
      WRITE(6,883)
  884 CONTINUE
      IF (NNODES(L) .EQ. 2) GO TO 6601
      IF (NNODES(L) .LT. 4) GO TO 5508
      WRITE(6,87) L,AAE(L),ELENTH(L),TYPE,MA(L),MB(L),MC(L),MD(L),
     1            STRNO(L,1),STRN90(L,1),STRN45P(L,1),STRN45N(L,1)
  986 IF (LOADS .EQ. 1) GO TO 6600
      DO 2211 K = 2,LOADS
      WRITE(6,94) STRNO(L,K),STRN90(L,K),STRN45P(L,K),STRN45N(L,K)
 2211 CONTINUE
      GO TO 6600
 5508 WRITE(6,88)L,AAE(L),ELENTH(L),TYPE,MA(L),MB(L),MC(L),STRNO(L,1),
     1          STRN90(L,1),STRN45P(L,1),STRN45N(L,1)
      GO TO 986
 6600 LINES = LINES + LOADS + 1
      WRITE(8,322) (AAE(L), L = 1,MEMBS)
  322 FORMAT(6F10.6)
 6601 DO 1503 KL=1,LOADS                                                         OPTSTAT
 1503 WRITE(6,1502)KL,ENGTOT(KL)                                                 OPTSTAT
 1502 FORMAT(///,20X,39HTHE TOTAL ENERGY FOR LOADING CONDITION ,I2,4H IS         OPTSTAT
     1 ,E12.4)                                                                   OPTSTAT
      LINES=1                                                                    OPTSTAT
      CALL PRNTDR(FR,DR,X,Y,Z,NN,MM,LOADS,JOINTS,NPAGE,NNMAX)                    OPTSTAT
   83 FORMAT(1X,4HMEMB,2X,5HTHICK,3X,4HAREA,2X,4HTYPE,1X,2HMA,2X,2HMB,           OPTSTAT
     12X,2HMC,2X,2HMD,4X,7HSIGMA-X,5X,7HSIGMA-Y,4X,8HSIGMA-XY,4X,                OPTSTAT
     27HESRATIO,6X,4HALS1,6X,4HALS2,1X,4HALS3,1X,4HALS4,1X,4HALS5,3X,           OPTSTAT
     26HENERGY)                                                                  OPTSTAT
   82 FORMAT(/I5,F7.3,F9.2,5I4,24X,E12.5,2E11.5,2X,4F5.2,E11.5)                  OPTSTAT
   85 FORMAT(43X,5H(LAM),1X,6H(THK0),3X,5H(AEX),2X,                              OPTSTAT
     17H(THK90),2X,5H(AEY))                                                    OPTSTAT
   90 FORMAT(65X,E12.5,E11.5,33X,E11.5)                                          OPTSTAT
   87 FORMAT(/I5, F7.3,F9.2,5I4,3E12.5,2E11.5,2X,4F5.2,E11.5)                    OPTSTAT
   88 FORMAT(/I5, F7.3,F9.2,4I4,4X,3E12.5,2E11.5,2X,4F5.2,E11.5)                 OPTSTAT
   89 FORMAT(/I5, F7.3,F9.2,3I4,8X,E12.5,24X,2E11.5,2X,F5.2,15X,E11.5)           OPTSTAT
   93 FORMAT(41X,E12.5,24X,E11.5,33X,E11.5)                                      OPTSTAT
   94 FORMAT(41X,3E12.5,E11.5,33X,E11.5)                                         OPTSTAT
  187 FORMAT(/I5,F7.3,F9.2,5I4,I4,4F8.5,2E11.5,2X,4F5.2,E11.5)                   OPTSTAT
  186 FORMAT(/I5,F7.3,F9.2,4I4,4X,I4,4F8.5,2E11.5,2X,4F5.2,E11.5)                OPTSTAT
  194 FORMAT(77X,E11.5,33X,E11.5)                                                OPTSTAT
   98 FORMAT(1H1,120X,5HPAGE ,I3/)                                               OPTSTAT
      IF (KANLYZE .EQ. 1) GO TO 45                                               OPTSTAT
```

```
      IF (LPRINT .EQ. 0) GO TO 45                                        OPTSTAT
      CALL LAYPR(LAM,NZDEG,NNDEG,NFDEG,LFLAG1,LFLAG2,                    OPTSTAT
     1NKIND,NCOUNT,NFAC)                                                 OPTSTAT
      KANLYZE = 1                                                        OPTSTAT
      DO 352 I = 1,MEMBS                                                 OPTSTAT
      AE(I) = AAE(I)                                                     OPTSTAT
  352 CONTINUE                                                           OPTSTAT
      DO 47 I = 1,NFAC                                                   OPTSTAT
      L = NKIND(I)                                                       OPTSTAT
      RNZDEG = NZDEG(L)                                                  OPTSTAT
      RLAM = LAM(L)                                                      OPTSTAT
      RNNDEG = NNDEG(L)                                                  OPTSTAT
      AEX(L) = RNZDEG/RLAM                                               OPTSTAT
      AEY(L) = RNNDEG/RLAM                                               OPTSTAT
      AE(L) = RLAM*THKLAM                                                OPTSTAT
   47 CONTINUE                                                           OPTSTAT
      GO TO 19                                                           OPTSTAT
   45 IF(KSTR.EQ.NSTR) GO TO 1000                                        OPTSTAT
      KSTR=KSTR+1                                                        OPTSTAT
      GO TO 1                                                            OPTSTAT
 1000 CONTINUE                                                           OPTSTAT
    2 FORMAT(14I5)                                                       OPTSTAT
    3 FORMAT(6F10.3)                                                     OPTSTAT
    5  FORMAT(1H1,///2X,10HBOUNDARIES ///)                               OPTSTAT
   16  FORMAT (3(F10.0,2I5))                                             OPTSTAT
 1009  FORMAT(1X,10I13)                                                  OPTSTAT
  116 FORMAT(////,5X,15HBASE AE(    ) = ,1PE14.6,5X,                   OPTSTAT
     125HWEIGHT OF THE STRUCTURE = ,1PE14.6,5X,                          OPTSTAT
     134HWEIGHT OF THE MEMBRANE ELEMENTS = ,E14.6)                       OPTSTAT
 143  FORMAT(   5X,13HSTRUCTURE NO= ,I5,9 X,12HNO OF LOADS= ,I5          OPTSTAT
     1,5X,11HCYCLE NO = ,I5,13X,34HWEIGHT OF THE SHEAR PANELS      = ,   OPTSTAT
     1E14.6/,83X,34HWEIGHT OF THE BAR ELEMENTS      = ,E14.6)            OPTSTAT
 147  FORMAT(   5X,10F12.6)                                              OPTSTAT
 148  FORMAT(// 5X,25HRELATIVE SIZES OF MEMBERS//)                       OPTSTAT
 149  FORMAT(52X,19HCYCLES IN SEARCH = ,I5)                              OPTSTAT
      STOP                                                               OPTSTAT
      END                                                                OPTSTAT
*DECK LMSIZE
      SUBROUTINE LMSIZE(AE,AEX,AEY,BASEAE,BASEA,ENGX,ENGY,ENGXY,ELENTH,  LMSIZE
     1LAM,MEMBS,TFR,AEXMIN,AEYMIN,AEXYMIN,MNLAYR)                        LMSIZE
      DIMENSION AE(1),AEX(1),AEY(1),ENGX(1),ENGY(1),ENGXY(1),ELENTH(1),  LMSIZE
     1LAM(1),TFR(6),AEXMIN(1),AEYMIN(1),AEXYMIN(1)                       LMSIZE
      DO 168 L=1,MEMBS                                                   LMSIZE
      IF(LAM(L) .EQ. 0)GO TO 168                                         LMSIZE
      AMAX=0.                                                            LMSIZE
      DO 167 I=1,3                                                       LMSIZE
      TFR(I)=0.                                                          LMSIZE
      IF(I-2)162,163,164                                                 LMSIZE
 162  AX=AEX(L)*ELENTH(L)                                                LMSIZE
      AY=ENGX(L)/BASEAE                                                  LMSIZE
      AZ=AEX(L)                                                          LMSIZE
      GO TO 165                                                          LMSIZE
 163  AX=AEY(L)*ELENTH(L)                                                LMSIZE
      AY=ENGY(L)/BASEAE                                                  LMSIZE
      AZ=AEY(L)                                                          LMSIZE
      GO TO 165                                                          LMSIZE
 164  AZ=(1.0  -AEX(L)-AEY(L))                                           LMSIZE
      AX=AZ*ELENTH(L)                                                    LMSIZE
      AY=ENGXY(L)/BASEAE                                                 LMSIZE
 165  IF(AZ .LT. 0.00001)GO TO 167                                       LMSIZE
  31  FORMAT(I5,10E12.4)                                                 LMSIZE
```

```
      TFR(I)=1000.*AZ*SQRT(AY/AX)                                        LMSIZE
      AMAX=AMAX+TFR(I)                                                   LMSIZE
  167 CONTINUE                                                           LMSIZE
      DO 166 I=1,3                                                       LMSIZE
  166 TFR(I)=TFR(I)/AMAX                                                 LMSIZE
      IF(AEX(L) .GT. 0.00001)AEX(L)=TFR(1)                               LMSIZE
      IF(AEY(L) .GT. 0.00001)AEY(L)=TFR(2)                               LMSIZE
  168 CONTINUE                                                           LMSIZE
      DO 170 L=1,MEMBS                                                   LMSIZE
      IF(LAM(L) .EQ. 0)GO TO 170                                         LMSIZE
      AX=AEX(L)*AE(L)*BASEA                                              LMSIZE
      AY=AEY(L)*AE(L)*BASEA                                              LMSIZE
      AZ=0.5*(1.-AEX(L)-AEY(L))*AE(L)*BASEA                              LMSIZE
      IF(AX .LT. AEXMIN(L))AX=AEXMIN(L)                                  LMSIZE
      IF(AY .LT. AEYMIN(L))AY=AEYMIN(L)                                  LMSIZE
      IF(AZ .LT. .5*AEXYMIN(L))AZ=.5*AEXYMIN(L)                                LMSIZE
      AE(L)=(AX+AY+2.*AZ)/BASEA                                          LMSIZE
      AEX(L)=AX/(AE(L)*BASEA)                                            LMSIZE
      AEY(L)=AY/(AE(L)*BASEA)                                            LMSIZE
  170 CONTINUE                                                           LMSIZE
      RETURN                                                             LMSIZE
      END                                                                LMSIZE
*DECK UNITEG
      SUBROUTINE UNITEG(EK,EGX,EGY,EGXY,EDR,EDR1,ELG1,EGST1,LD,KDQ,MDQ,  UNITEG
     1NDMY,NACT,NND,S,STRG,II,INDX)                                      UNITEG
      DIMENSION EK(12,12),EGST1(NACT,LD),       EDR(12,LD),EDR1(8,NACT), UNITEG
     1ELG1(1),          KDQ(1),MDQ(NACT,LD),S(12,LD)                     UNITEG
      IF(NND .EQ. 2) GO TO 10                                            UNITEG
      KX=8                                                               UNITEG
      IF(NND .EQ. 3) KX=6                                                UNITEG
      DO 9 K=1,LD                                                        UNITEG
       DO 8 I=1,KX                                                       UNITEG
      S(I,K)=0.                                                          UNITEG
      DO 8 J=1,KX                                                        UNITEG
    8 S(I,K)=S(I,K)+EK(I,J)*EDR(J,K)                                     UNITEG
    9 CONTINUE                                                           UNITEG
   10 ADR=0.                                                             UNITEG
      DO 20 I=1,NDMY                                                     UNITEG
      KF=KDQ(I)                                                          UNITEG
      IF(KF .EQ. 0) GO TO 20                                             UNITEG
      DO 19 J=1,KF                                                       UNITEG
      KX=MDQ(I,J)                                                        UNITEG
      KY=1                                                               UNITEG
      IF(KX .GT. 0) GO TO 17                                             UNITEG
      KX=-KX                                                             UNITEG
      KY=-1                                                              UNITEG
   17 ELG1(J)=0.                                                         UNITEG
      KH=8                                                               UNITEG
      IF(NND .EQ. 3) KH=6                                                UNITEG
      IF(NND .EQ. 2) KH=2                                                UNITEG
      DO 18 K=1,KH                                                       UNITEG
   18 ELG1(J)=ELG1(J)+S(K,KX)*EDR1(K,I)                                  UNITEG
      ELG1(J)=ELG1(J)*KY                                                 UNITEG
   19 IF(ELG1(J) .GE. 0.) ADR=ADR+ELG1(J)/EGST1(I,J)                     UNITEG
   20 CONTINUE                                                           UNITEG
      STRG=ADR                                                           UNITEG
      IF(INDX .EQ. 0) RETURN                                             UNITEG
      IF(II .EQ. 1) EGX=ADR                                              UNITEG
      IF(II .EQ. 2) EGY=ADR                                              UNITEG
      IF(II .EQ. 3) EGXY=ADR                                             UNITEG
      IF(II .EQ. 4) EGXY=EGXY+ADR                                        UNITEG
```

```
      RETURN                                                            UNITEG
      END                                                               UNITEG
*DECK PREPAR
      SUBROUTINE PREPAR(AE,AEX,AEY,ALS,ALSTRS,AX,AY,AZ,BA,BAE,EEK,E1,E2, PREPAR
     1SM,PMU,ELCNST,ESRTIO,ELENG,ISO,NISO,KX,KY,LAM,LAYERD,LD,MYG,YGM,   PREPAR
     2POISON,NND,TFR,TFFR,INDEX)                                        PREPAR
      DIMENSION ALS(5),ALSTRS(1),EEK(8,8),ELCNST(1),ESRTIO(1),          PREPAR
     1ELENG(1),YGM(1),TFR(1),TFFR(4),POISON(1)                          PREPAR
      KX=MYG                                                            PREPAR
      E1=YGM(KX)                                                        PREPAR
      E2=E1                                                             PREPAR
      PMU=POISON(KX)                                                    PREPAR
      SM=E1/(2.*(1.+PMU))                                               PREPAR
      IF(INDEX .EQ. 0)GO TO 20                                          PREPAR
      AX=1.                                                             PREPAR
      AY=0.                                                             PREPAR
      AZ=0.                                                             PREPAR
      DO 174 J=1,LD                                                     PREPAR
      ELENG(J)=0.                                                       PREPAR
 174  ESRTIO(J)=0.                                                      PREPAR
      KY=5*(KX-1)                                                       PREPAR
      DO 1504 I=1,5                                                     PREPAR
      KY=KY+1                                                           PREPAR
1504  ALS(I)=ALSTRS(KY)*BAE                                             PREPAR
      TFR(1)=ALS(1)                                                     PREPAR
      DO 216 I=2,5                                                      PREPAR
 216  TFR(I)=ALS(I)/ALS(1)                                              PREPAR
      IF (LAM .EQ. 0) GO TO 20                                          PREPAR
      ALS(3)=100000.*BAE                                                PREPAR
      ALS(5)=ALS(3)                                                     PREPAR
      ALS(4) = ALS(3)                                                   PREPAR
  20  IF(NND .EQ. 2)GO TO 400                                           PREPAR
      KX=0                                                              PREPAR
      IF(NISO .GT. 0)KX=MYG-ISO                                         PREPAR
      IF(KX .LE. 0)GO TO 65                                             PREPAR
      KY=2*(KX-1)+1                                                     PREPAR
      E2=ELCNST(KY)                                                     PREPAR
      SM=ELCNST(KY+1)                                                   PREPAR
  65  CONTINUE                                                          PREPAR
      TFFR(1)=AE    *BA                                                 PREPAR
      KX=4                                                              PREPAR
      IF(LAYERD .GT. 0)GO TO 171                                        PREPAR
      KX=1                                                              PREPAR
      GO TO 400                                                         PREPAR
 171  IF(LAM .EQ. 0)KX=1                                                PREPAR
      IF(MYG .LE. ISO)KX=1                                              PREPAR
      IF(KX .LE. 1)GO TO 400                                            PREPAR
      IF(INDEX .EQ. 1)GO TO 150                                         PREPAR
      KY=2*NND                                                          PREPAR
      IF(NND .GT. 4)KY=8                                                PREPAR
      DO 151 I=1,KY                                                     PREPAR
      DO 151 J=1,KY                                                     PREPAR
 151  EEK(I,J)=0.                                                       PREPAR
 150  TFFR(1)=AEX*AE*BA                                                 PREPAR
      TFFR(2)=AEY*AE*BA                                                 PREPAR
      TFFR(3)=AE*BA*(1.-AEX-AEY)/2.                                     PREPAR
      TFFR(4)=TFFR(3)                                                   PREPAR
 400  IF(INDEX .EQ. 0)RETURN                                            PREPAR
      E1=E1*10.**6                                                      PREPAR
      E2 =E2*10.**6                                                     PREPAR
      SM=SM*10.**6                                                      PREPAR
```

```
      RETURN                                                            PREPAR
      END                                                               PREPAR
*DECK GAUSS1
      SUBROUTINE GAUSS1(A,    D,IC,ID,L,N,NN)                           GAUSS1
      DIMENSION A(1),IC(1),ID(1),       D(NN,L)                         GAUSS1
      DO 40 K=1,L                                                       GAUSS1
      DO 30 I=1,N                                                       GAUSS1
      I1=I-1                                                            GAUSS1
      IF(I1 .EQ. 0) GO TO 30                                            GAUSS1
      DO 20 J=1,I1                                                      GAUSS1
      IF(IC(I) .GT. J)GO TO 20                                          GAUSS1
      IX=ID(I)-I+J                                                      GAUSS1
      D(I,K)=D(I,K)-A(IX)*D(J,K)                                        GAUSS1
   20 CONTINUE                                                          GAUSS1
   30 CONTINUE                                                          GAUSS1
   40 CONTINUE                                                          GAUSS1
      DO 70 I=1,N                                                       GAUSS1
      KX=ID(I)                                                          GAUSS1
      DO 70 K=1,L                                                       GAUSS1
   70 D(I,K)=D(I,K)/A(KX)                                               GAUSS1
      DO 90 K=1,L                                                       GAUSS1
      IX=N                                                              GAUSS1
      DO 90 I=2,N                                                       GAUSS1
      IX=IX-1                                                           GAUSS1
      I1=I-1                                                            GAUSS1
      KX=IX                                                             GAUSS1
      DO 80 J=1,I1                                                      GAUSS1
      KX=KX+1                                                           GAUSS1
      IF(IC(KX) .GT. IX)GO TO 80                                        GAUSS1
      KY=ID(KX)-KX+IX                                                   GAUSS1
      D(IX,K)=D(IX,K)-A(KY)*D(KX,K)                                     GAUSS1
   80 CONTINUE                                                          GAUSS1
   90 CONTINUE                                                          GAUSS1
      GO TO 110                                                         GAUSS1
  100 WRITE(6,120)                                                      GAUSS1
  120 FORMAT(///2X,21HSTRUCTURE IS UNSTABLE///)                         GAUSS1
  110 RETURN                                                            GAUSS1
      END                                                               GAUSS1
*DECK POP
      SUBROUTINE POP(MMB,JN,MM,MA,MB,MC,MD,KTYPE,IC,ID,NZ,NR)           POP
      DIMENSION MA(1),MB(1),MC(1),MD(1),IC(1),ID(1),KTYPE(1)            POP
      IX(I,J)=I*(J-1)+1                                                 POP
      NZ=0                                                              POP
      NN=MM*JN                                                          POP
      NET=0                                                             POP
      DO 10 I=1,NN                                                      POP
   10 IC(I)=NN                                                          POP
      DO 50 L=1,MMB                                                     POP
      NNODES=2                                                          POP
      ITRI=0                                                            POP
      KX=IX(MM,MA(L))                                                   POP
      KY=IX(MM,MB(L))                                                   POP
   15 IF(IC(KY) .LT. KX) GO TO 18                                       POP
      DO 19 I=1,MM                                                      POP
      IC(KY)=KX                                                         POP
   19 KY=KY+1                                                           POP
   18 IF(KTYPE(L)-3)20,16,17                                            POP
   16 IF(ITRI .EQ. 1)GO TO 20                                           POP
      KY=IX(MM,MC(L))                                                   POP
      ITRI=1                                                            POP
      NNODES=3                                                          POP
```

```
      GO TO 15                                                          POP
   17 IF(ITRI .EQ. 2)GO TO 20                                           POP
      IF(ITRI .EQ. 1)GO TO 14                                           POP
      KY=IX(MM,MC(L))                                                   POP
      ITRI=ITRI+1                                                       POP
      NNODES=4                                                          POP
      GO TO 15                                                          POP
   14 KY=IX(MM,MD(L))                                                   POP
      ITRI=ITRI+1                                                       POP
      GO TO 15                                                          POP
   20 NET=NET+(MM**2)*((NNODES*(NNODES-1))/2)                           POP
   50 CONTINUE                                                          POP
      NET=NET-(MM**2)*NR                                                POP
      DO 30 I=1,NN,MM                                                   POP
      IF(IC(I) .LT. I)GO TO 30                                          POP
      KX=I                                                              POP
      DO 25 J=1,MM                                                      POP
      IC(KX)=I                                                          POP
   25 KX=KX+1                                                           POP
   30 CONTINUE                                                          POP
      DO 40 I=1,NN                                                      POP
      NZ=NZ+(I-IC(I)+1)                                                 POP
   40 ID(I)=NZ                                                          POP
      KX=(NN*(NN+1))/2                                                  POP
      NET=NET+(MM*(MM+1)*JN)/2                                          POP
      WRITE(6,2)                                                        POP
      WRITE(6,3) KX,NET,NZ                                              POP
      WRITE(6,4)                                                        POP
      WRITE(6,5)(IC(I),I=1,NN)                                          POP
      WRITE(6,6)                                                        POP
      WRITE(6,5)(ID(I),I=1,NN)                                          POP
    2 FORMAT(1H1,////20X,16HGROSS POPULATION,4X,14HNET POPULATION,      POP
     14X,19HAPPARENT POPULATION///)                                     POP
    3 FORMAT(18X,I14,I18,I22//)                                         POP
    4 FORMAT(//2X,36HSTARTING ROW NUMBERS FOR EACH COLUMN///)           POP
    5 FORMAT(5X,10I12)                                                  POP
    6 FORMAT(//2X,62HNUMBERS OF DIAGONAL ELEMENTS IN SINGLE ARRAY STIFFN PO
     1ESS MATRIX ///)                                                   PO
      RETURN                                                            POP
      END                                                               POP
*DECK COORD
      SUBROUTINE COORD(K1,K2,K3,K4,X,Y,Z,AA,XI,ETA,AL,NND,NO)           COORD
      DIMENSION X(1),Y(1),Z(1),AA(3,3),AB(3),XI(5),ETA(5)               COORD
      XCOMP=X(K2)-X(K1)                                                 COORD
      YCOMP=Y(K2)-Y(K1)                                                 COORD
      ZCOMP=Z(K2)-Z(K1)                                                 COORD
      AL=SQRT(XCOMP**2+YCOMP**2+ZCOMP**2)                               COORD
      AA(1,1)=XCOMP/AL                                                  COORD
      AA(1,2)=YCOMP/AL                                                  COORD
      AA(1,3)=ZCOMP/AL                                                  COORD
      IF(NND .LT. 3)RETURN                                              COORD
      XCOMP=X(K3)-X(K1)                                                 COORD
      YCOMP=Y(K3)-Y(K1)                                                 COORD
      ZCOMP=Z(K3)-Z(K1)                                                 COORD
      AL=SQRT(XCOMP**2+YCOMP**2+ZCOMP**2)                               COORD
      AB(1)=XCOMP/AL                                                    COORD
      AB(2)=YCOMP/AL                                                    COORD
      AB(3)=ZCOMP/AL                                                    COORD
      AL=SQRT((AA(1,2)*AB(3)-AA(1,3)*AB(2))**2+(AA(1,3)*AB(1)           COORD
     1-AA(1,1)*AB(3))**2+(AA(1,1)*AB(2)-AA(1,2)*AB(1))**2)              COORD
      AA(2,1)=((AA(1,3)**2)*AB(1)-AA(1,1)*AA(1,3)*AB(3)-AA(1,1)*        COORD
```

```
     1AA(1,2)*AB(2)+(AA(1,2)**2)*AB(1))/AL                                  COORD
      AA(2,2)=((AA(1,1)**2)*AB(2)-AA(1,1)*AA(1,2)*AB(1)-AA(1,2)*            COORD
     2AA(1,3)*AB(3)+(AA(1,3)**2)*AB(2))/AL                                  COORD
      AA(2,3)=((AA(1,2)**2)*AB(3)-AA(1,2)*AA(1,3)*AB(2)-AA(1,1)*            COORD
     3AA(1,3)*AB(1)+(AA(1,1)**2)*AB(3))/AL                                  COORD
      IF(NO .EQ. 1)RETURN                                                   COORD
      XI(1)=0.0                                                             COORD
      ETA(1)=0.0                                                            COORD
      XI(2)=(X(K2)-X(K1))*AA(1,1)+(Y(K2)-Y(K1))*AA(1,2)+(Z(K2)-Z(K1))*AA    COORD
     1(1,3)                                                                 COORD
      ETA(2)=0.0                                                            COORD
      XI(3)=(X(K3)-X(K1))*AA(1,1)+(Y(K3)-Y(K1))*AA(1,2)+(Z(K3)-Z(K1))*AA    COORD
     1(1,3)                                                                 COORD
      ETA(3)=(X(K3)-X(K1))*AA(2,1)+(Y(K3)-Y(K1))*AA(2,2)+(Z(K3)-Z(K1))*A    COORD
     1A(2,3)                                                                COORD
      IF(NND .LE. 3)RETURN                                                  COORD
      XI(4)=(X(K4)-X(K1))*AA(1,1)+(Y(K4)-Y(K1))*AA(1,2)+(Z(K4)-Z(K1))*AA    COORD
     1(1,3)                                                                 COORD
      ETA(4)=(X(K4)-X(K1))*AA(2,1)+(Y(K4)-Y(K1))*AA(2,2)+(Z(K4)-Z(K1))*A    COORD
     1A(2,3)                                                                COORD
      XI(5)=(XI(2)+XI(3)+XI(4)) /4.0                                        COORD
      ETA(5)=(ETA(3)+ETA(4))/4.0                                            COORD
      RETURN                                                                COORD
      END                                                                   COORD
*DECK QDRLTL
      SUBROUTINE QDRLTL(EK,EKK,TH,QUAD,MA,MB,MC,MD,MAA,MBB,MCC,XI,ETA,      QDRLTL
     1NNODES,EE,TRANG,NO)                                                   QDRLTL
      DIMENSION EK(12,12),EKK(12,12),MAA(1),MBB(1),MCC(1),XI(5),ETA(5)      QDRLTL
     1,EE(3,3) ,TRANG(1)                                                    QDRLTL
      DO 125 I=1,12                                                         QDRLTL
      DO 125 J=1,12                                                         QDRLTL
 125  EK(I,J)=0.                                                            QDRLTL
      NNRM=0                                                                QDRLTL
      SHR=1.0                                                               QDRLTL
      IF(NNODES     .LE. 4)GO TO 108                                        QDRLTL
      NNRM=1                                                                QDRLTL
      IF(NNODES     .EQ. 5)GO TO 108                                        QDRLTL
      IF(NNODES     - 7)104,105,106                                         QDRLTL
 104  XCOMP=XI(3)-XI(2)                                                     QDRLTL
      YCOMP=ETA(3)-ETA(2)                                                   QDRLTL
      GO TO 107                                                             QDRLTL
 105  XCOMP=XI(4)-XI(3)                                                     QDRLTL
      YCOMP=ETA(4)-ETA(3)                                                   QDRLTL
      GO TO 107                                                             QDRLTL
 106  XCOMP=XI(4)-XI(1)                                                     QDRLTL
      YCOMP=ETA(4)-ETA(1)                                                   QDRLTL
 107  ALL=SQRT(XCOMP**2+YCOMP**2)                                           QDRLTL
      SHR=XCOMP/ALL                                                         QDRLTL
 108  QUAD=0.                                                               QDRLTL
      DO 130 I=1,4                                                          QDRLTL
      CALL PLSTIF(EKK,TH,TRIANG,MAA(I),MBB(I),MCC(I),XI,ETA,EE,SHR,NNRM)    QDRLTL
      QUAD=QUAD+TRIANG                                                      QDRLTL
      TRANG(I)=TRIANG                                                       QDRLTL
 130  CALL SUM(EK,EKK,MAA(I),MBB(I),MCC(I))                                 QDRLTL
      CALL CONDNS(EK,EKK,MA,MB,MC,MD,NO)                                    QDRLTL
      RETURN                                                                QDRLTL
      END                                                                   QDRLTL
*DECK PLSTIF
      SUBROUTINE PLSTIF(EKK,TH,TRIANG,MA,MB,MC,X,Y,EE,SHR,NONORM)           PLSTIF
      DIMENSION EKK(12,12),X(1),Y(1),EE(3,3),                               PLSTIF
     1     U(6),            A(3,3),E1(3),E2(3),AX(3)                       PLSTIF
```

```
      CALL CRAMER(A,TRIANG,X,Y,MA,MB,MC)                                PLSTIF
      DO 20 I = 1,6                                                     PLSTIF
      DO 15 II = 1,6                                                    PLSTIF
   15 U(II) = 0.0                                                       PLSTIF
      U(I) = 1.0                                                        PLSTIF
      E1(1) = A(1,1)*U(1) + A(1,2)*U(3) + A(1,3)*U(5)                   PLSTIF
      E1(2) = A(2,1)*U(2) + A(2,2)*U(4) + A(2,3)*U(6)                   PLSTIF
      E1(3) = A(1,1)*U(2) + A(1,2)*U(4) + A(1,3)*U(6) + A(2,1)*U(1) +   PLSTIF
     1        A(2,2)*U(3) + A(2,3)*U(5)                                  PLSTIF
      DO 20 J = I,6                                                     PLSTIF
      DO 16 II = 1,6                                                    PLSTIF
   16 U(II) = 0.0                                                       PLSTIF
      U(J) = 1.0                                                        PLSTIF
      E2(1) = A(1,1)*U(1) + A(1,2)*U(3) + A(1,3)*U(5)                   PLSTIF
      E2(2) = A(2,1)*U(2) + A(2,2)*U(4) + A(2,3)*U(6)                   PLSTIF
      E2(3) = A(1,1)*U(2) + A(1,2)*U(4) + A(1,3)*U(6) + A(2,1)*U(1) +   PLSTIF
     1        A(2,2)*U(3) + A(2,3)*U(5)                                  PLSTIF
      EKK(I,J) = 0.0                                                    PLSTIF
      IF (NONORM .EQ. 0) GO TO 14                                       PLSTIF
      AX(1)=SHR**2                                                      PLSTIF
      AX(2)=2.*AX(1)-1.                                                 PLSTIF
      AX(1)=2.*SQRT((1.-AX(1))*AX(1))                                   PLSTIF
      E1(3)=(E1(2)-E1(1))*AX(1)+E1(3)*AX(2)                             PLSTIF
      E2(3)=(E2(2)-E2(1))*AX(1)+E2(3)*AX(2)                             PLSTIF
      E1(1) = 0.0                                                       PLSTIF
      E1(2) = 0.0                                                       PLSTIF
      E2(1) = 0.0                                                       PLSTIF
      E2(2) = 0.0                                                       PLSTIF
   14 DO 18 K = 1,3                                                     PLSTIF
      AX(K) = 0.0                                                       PLSTIF
      DO 17 L = 1,3                                                     PLSTIF
   17 AX(K) = AX(K) + EE(K,L)*E2(L)                                     PLSTIF
   18 CONTINUE                                                          PLSTIF
      DO 19 K = 1,3                                                     PLSTIF
   19 EKK(I,J) = EKK(I,J) + E1(K)*AX(K)                                 PLSTIF
      EKK(I,J) = EKK(I,J)*TH*TRIANG                                     PLSTIF
   20 CONTINUE                                                          PLSTIF
      DO 30 I = 1,5                                                     PLSTIF
      IX = I + 1                                                        PLSTIF
      DO 30 J = IX,6                                                    PLSTIF
   30 EKK(J,I) = EKK(I,J)                                               PLSTIF
      RETURN                                                            PLSTIF
      END                                                               PLSTIF
*DECK CRAMER
      SUBROUTINE CRAMER(A,TRIANG,X,Y,MA,MB,MC)                          CRAMER
      DIMENSION A(3,3),X(1),Y(1)                                        CRAMER
      TRIANG = X(MA)*(Y(MB) - Y(MC)) -Y(MA)*(X(MB) - X(MC)) +           CRAMER
     1         (X(MB)*Y(MC) - X(MC)*Y(MB))                              CRAMER
      A(1,1) = Y(MB) - Y(MC)                                            CRAMER
      A(2,1) = X(MC) - X(MB)                                            CRAMER
      A(3,1) = X(MB)*Y(MC) - X(MC)*Y(MB)                                CRAMER
      A(1,2) = Y(MC) - Y(MA)                                            CRAMER
      A(2,2) = X(MA) - X(MC)                                            CRAMER
      A(3,2) = X(MC)*Y(MA) - X(MA)*Y(MC)                                CRAMER
      A(1,3) = Y(MA) - Y(MB)                                            CRAMER
      A(2,3) = X(MB) - X(MA)                                            CRAMER
      A(3,3) = X(MA)*Y(MB) - X(MB)*Y(MA)                                CRAMER
      DO 10 I = 1,3                                                     CRAMER
      DO 10 J = 1,3                                                     CRAMER
   10 A(I,J) = A(I,J)/TRIANG                                            CRAMER
      TRIANG =(ABS(TRIANG))/2.0                                         CRAMER
```

```
      RETURN                                                        CRAMER
      END                                                           CRAMER
*DECK SUM
      SUBROUTINE SUM(EK,EKK,MA,MB,MC)                               SUM
      DIMENSION EK(12,12),EKK(12,12),NA(3)                          SUM
      M=2                                                           SUM
      NA(1)=2*(MA-1)+1                                              SUM
      NA(2)=2*(MB-1)+1                                              SUM
      NA(3)=2*(MC-1)+1                                              SUM
      IH=0                                                          SUM
      DO 100 I=1,6                                                  SUM
      JH=0                                                          SUM
      IF(I .LE. IH)GO TO 30                                         SUM
      IH=IH+M                                                       SUM
      IHH=IH/M                                                      SUM
      KX=NA(IHH)                                                    SUM
   30 DO 90 J=1,6                                                   SUM
      IF(J .LE. JH)GO TO 60                                         SUM
      JH=JH+M                                                       SUM
      IHH=JH/M                                                      SUM
      KY=NA(IHH)                                                    SUM
```

```
   60  EK(KX,KY)=EK(KX,KY)+EKK(I,J)                                    SUM
   90  KY=KY+1                                                         SUM
  100  KX=KX+1                                                         SUM
       RETURN                                                          SUM
       END                                                             SUM
*DECK CONDNS
       SUBROUTINE CONDNS(EK,EKK,MA,MB,MC,MD,NO)                        CONDNS
       DIMENSION EK(12,12),EKK(12,12)                                  CONDNS
       DO 5 I=1,8                                                      CONDNS
       DO 5 J=1,8                                                      CONDNS
    5  EKK(I,J)=0.                                                     CONDNS
       DET=EK(9,9)*EK(10,10)-EK(9,10)**2                               CONDNS
       AX=EK(9,9)                                                      CONDNS
       EK(9,9)=EK(10,10)/DET                                           CONDNS
       EK(10,10)=AX/DET                                                CONDNS
       EK(9,10)=-EK(9,10)/DET                                          CONDNS
       EK(10,9)=EK(9,10)                                               CONDNS
       KX=0                                                            CONDNS
       DO 10 I=9,10                                                    CONDNS
       KX=KX+1                                                         CONDNS
       DO 10 J=1,8                                                     CONDNS
       DO 10 K=9,10                                                    CONDNS
   10  EKK(KX,J)=EKK(KX,J)+EK(I,K)*EK(K,J)                             CONDNS
       IF(NO .EQ. 1)RETURN                                             CONDNS
       KX=0                                                            CONDNS
       DO 20 I=9,10                                                    CONDNS
       KX=KX+1                                                         CONDNS
       DO 20 J=1,8                                                     CONDNS
       EK(I,J)=EKK(KX,J)                                               CONDNS
   20  EKK(KX,J)=0                                                     CONDNS
       DO 30 I=1,8                                                     CONDNS
       DO 30 J=1,8                                                     CONDNS
       DO 30 K=9,10                                                    CONDNS
   30  EKK(I,J)=EKK(I,J)+EK(I,K)*EK(K,J)                               CONDNS
       DO 40 I=1,8                                                     CONDNS
       DO 40 J=1,8                                                     CONDNS
   40  EK(I,J)=EK(I,J)-EKK(I,J)                                        CONDNS
       IF(NO .EQ. 2)RETURN                                             CONDNS
       IF(MC .LT. MB)CALL CHANGE(EK,3,5,4,12,12,0)                     CONDNS
       IF(MD .LT. MB)CALL CHANGE(EK,3,7,4,12,12,0)                     CONDNS
       IF(MD .LT. MC)CALL CHANGE(EK,5,7,4,12,12,0)                     CONDNS
       RETURN                                                          CONDNS
       END                                                             CONDNS
*DECK CHANGE
       SUBROUTINE CHANGE(EK,IX,IY,NND,M,L,IR)                          CHANGE
       DIMENSION EK( M, L)                                             CHANGE
       KX=IX                                                           CHANGE
       KY=IY                                                           CHANGE
       M2=2*NND                                                        CHANGE
       IF(IR .EQ. 1)M2=L                                               CHANGE
       DO 10 I=1,2                                                     CHANGE
       DO 5 J=1,M2                                                     CHANGE
       AX=EK(KX,J)                                                     CHANGE
       EK(KX,J)=EK(KY,J)                                               CHANGE
    5  EK(KY,J)=AX                                                     CHANGE
       KX=KX+1                                                         CHANGE
   10  KY=KY+1                                                         CHANGE
       IF(IR .EQ. 1)RETURN                                             CHANGE
       KX=KX-2                                                         CHANGE
       KY=KY-2                                                         CHANGE
       DO 20 I=1,2                                                     CHANGE
```

```
      DO 15 J=1,M2                                                    CHANGE
      AX=EK(J,KX)                                                     CHANGE
      EK(J,KX)=EK(J,KY)                                               CHANGE
   15 EK(J,KY)=AX                                                     CHANGE
      KX=KX+1                                                         CHANGE
   20 KY=KY+1                                                         CHANGE
      RETURN                                                          CHANGE
      END                                                             CHANGE
*DECK TRNSFM
      SUBROUTINE TRNSFM(EK,AA,B,C,MM,NND,M)                           TRNSFM
      DIMENSION EK(12,12),AA(3,3),B( M, M),C( M, M)                  TRNSFM
      M2=2*NND                                                        TRNSFM
      IF(NND .GT. 4)M2=8                                              TRNSFM
      M3=MM*NND                                                       TRNSFM
      IF(NND .GT. 4)M3=4*MM                                           TRNSFM
      DO 100 I=1,M2                                                   TRNSFM
      JA=MM                                                           TRNSFM
      KA=0                                                            TRNSFM
      IA=0                                                            TRNSFM
      DO 100 J=1,M3                                                   TRNSFM
      B(I,J)=0.0                                                      TRNSFM
      IF(J-JA)90,90,80                                                TRNSFM
   80 JA=JA+MM                                                        TRNSFM
      KA=KA+2                                                         TRNSFM
      IA=IA+MM                                                        TRNSFM
   90 JAA=J-IA                                                        TRNSFM
      DO 100 K=1,2                                                    TRNSFM
      KAA=K+KA                                                        TRNSFM
  100 B(I,J)=B(I,J)+EK(I,KAA)*AA(K,JAA)                               TRNSFM
      DO 200 J=1,M3                                                   TRNSFM
      JA=MM                                                           TRNSFM
      KA=0                                                            TRNSFM
      IA=0                                                            TRNSFM
      DO 200 I=1,M3                                                   TRNSFM
      C(I,J)=0.0                                                      TRNSFM
      IF(I-JA)190,190,180                                             TRNSFM
  180 JA=JA+MM                                                        TRNSFM
      KA=KA+2                                                         TRNSFM
      IA=IA+MM                                                        TRNSFM
  190 JAA=I-IA                                                        TRNSFM
      DO 200 K=1,2                                                    TRNSFM
      KAA=K+KA                                                        TRNSFM
  200 C(I,J)=C(I,J)+AA(K,JAA)*B(KAA,J)                                TRNSFM
      RETURN                                                          TRNSFM
      END                                                             TRNSFM
*DECK ELSTIF
      SUBROUTINE ELSTIF(A,B,C,AE,MM,AL,E)                             ELSTIF
      DIMENSION A(3,3),B(12,12),C(12,12)                              ELSTIF
      DO 100 I = 1,6                                                  ELSTIF
      DO 100 J = 1,6                                                  ELSTIF
      C(I,J) = 0.0                                                    ELSTIF
  100 CONTINUE                                                        ELSTIF
      EK=(AE*E)/AL                                                    ELSTIF
      DO 25 I=1,MM                                                    ELSTIF
      J=I+MM                                                          ELSTIF
      B(1,I)=EK*A(1,I)                                                ELSTIF
      B(1,J)= -B(1,I)                                                 ELSTIF
      B(2,I) =-B(1,I)                                                 ELSTIF
   25 B(2,J)=B(1,I)                                                   ELSTIF
      DO 26 I=1,MM                                                    ELSTIF
      DO 26 J=1,MM                                                    ELSTIF
```

```
     26  C(I,J)=A(1,I)*B(1,J)                                          ELSTIF
         DO 36 I=1,MM                                                  ELSTIF
         I1=I+MM                                                       ELSTIF
         DO 36 J=1,MM                                                  ELSTIF
         J1=J+MM                                                       ELSTIF
         C(I,J1)=-C(I,J)                                               ELSTIF
         C(J1,I)=-C(I,J)                                               ELSTIF
     36  C(I1,J1)=C(I,J)                                               ELSTIF
         RETURN                                                        ELSTIF
         END                                                           ELSTIF
*DECK ASEMBL
         SUBROUTINE ASEMBL(A,B,MA,MB,MC,MD,MM,ID,NNODES,M)             ASEMBL
         DIMENSION A(1),B(M,M),ID(1),NA(4),NAA(3)                      ASEMBL
         IX(I,J)=I*(J-1)+1                                             ASEMBL
         NND=NNODES                                                    ASEMBL
         IF(NND .GT. 4)NND=4                                           ASEMBL
         M2=NND*MM                                                     ASEMBL
         NA(1)=IX(MM,MA)                                               ASEMBL
         NA(2)=IX(MM,MB)                                               ASEMBL
         IF(NNODES .GE. 3)NA(3)=IX(MM,MC)                              ASEMBL
         IF(NNODES .GE. 4)NA(4)=IX(MM,MD)                              ASEMBL
         IF(NNODES .LE. 3)GO TO 5                                      ASEMBL
         DO 4 I=1,3                                                    ASEMBL
         KX=I/3                                                        ASEMBL
         KY=I/2                                                        ASEMBL
         IF(NA(KX+2) .LT. NA(KY+3))GO TO 4                             ASEMBL
         KH=NA(KX+2)                                                   ASEMBL
         NA(KX+2)=NA(KY+3)                                             ASEMBL
         NA(KY+3)=KH                                                   ASEMBL
      4  CONTINUE                                                      ASEMBL
      5  DO 10 I=2,NND                                                 ASEMBL
     10  NAA(I-1)=NA(I)-NA(I-1)-MM                                     ASEMBL
         KH=MM                                                         ASEMBL
         IAA=NA(1)                                                     ASEMBL
         KHH=1                                                         ASEMBL
         DO 30 J=1,M2                                                  ASEMBL
         IF(J .LE. KH)GO TO 15                                         ASEMBL
         KHH=KHH+1                                                     ASEMBL
         IAA=NA(KHH)                                                   ASEMBL
         KH=KH+MM                                                      ASEMBL
     15  JX=ID(IAA)-IAA+NA(1)                                          ASEMBL
         KY=MM                                                         ASEMBL
         DO 25 I=1,J                                                   ASEMBL
         IF(J .LE.KY .OR. I .LE. KY)GO TO 20                           ASEMBL
         KX=I/MM                                                       ASEMBL
         JX=JX+NAA(KX)                                                 ASEMBL
         KY=KY+MM                                                      ASEMBL
     20  A(JX)=A(JX)+B(I,J)                                            ASEMBL
     25  JX=JX+1                                                       ASEMBL
     30  IAA=IAA+1                                                     ASEMBL
         RETURN                                                        ASEMBL
         END                                                           ASEMBL
*DECK PRINTK
         SUBROUTINE PRINTK(SK,IDIAG,NN)                                PRINTK
         DIMENSION SK(1),IDIAG(1)                                      PRINTK
         DO 80 I=1,NN                                                  PRINTK
         IF(I .GT. 1) GO TO 65                                         PRINTK
         KX=1                                                          PRINTK
         KY=1                                                          PRINTK
         GO TO 70                                                      PRINTK
     65  KX=IDIAG(I-1)+1                                               PRINTK
```

```
      KY=IDIAG(I)                                                    PRINTK
   70 WRITE(6,3)I                                                    PRINTK
   80 WRITE(6,2)(SK(K),K=KX,KY)                                      PRINTK
    3 FORMAT(I4)                                                     PRINTK
    2 FORMAT(10X,10E12.4)                                            PRINTK
      RETURN                                                         PRINTK
      END                                                            PRINTK
*DECK REDUCE
      SUBROUTINE REDUCE (F,IB,N,NB,L,NN)                             REDUCE
      DIMENSION F(NN,L) ,IB(1)                                       REDUCE
      DO 5 J=1,L                                                     REDUCE
      IH=NB                                                          REDUCE
      NH=N                                                           REDUCE
1     I=IB(IH)                                                       REDUCE
      IF(I-NH) 2,4,4                                                 REDUCE
2     NH1=NH-1                                                       REDUCE
      DO 3 K=I,NH1                                                   REDUCE
      K1=K+1                                                         REDUCE
3     F(K,J) =F(K1,J)                                                REDUCE
4     IH=IH-1                                                        REDUCE
      NH=NH-1                                                        REDUCE
      IF(IH.EQ.0) GO TO 5                                            REDUCE
      GO TO1                                                         REDUCE
5     CONTINUE                                                       REDUCE
      RETURN                                                         REDUCE
      END                                                            REDUCE
*DECK BOUND2
      SUBROUTINE BOUND2(A,IB,N,NB,ID,IC)                             BOUND2
      DIMENSION A(1),IB(1),ID(1),IC(1)                               BOUND2
      IH=NB                                                          BOUND2
      NH=N                                                           BOUND2
      DO 30 JA=1,NB                                                  BOUND2
      IA=IB(IH)                                                      BOUND2
      IF(IA .GE. NH) GO TO 20                                        BOUND2
      KH=IA+1                                                        BOUND2
      IF(IA .GT. 1) GO TO 5                                          BOUND2
      KX=1                                                           BOUND2
      JX=1                                                           BOUND2
      GO TO 6                                                        BOUND2
   5  JX=ID(IA)-ID(IA-1)                                             BOUND2
      KX=ID(IA-1)+1                                                  BOUND2
   6  DO 10 I=KH,NH                                                  BOUND2
      KY=1                                                           BOUND2
      IF(IC(I) .LE. IA) GO TO 7                                      BOUND2
      IC(I-1)=IC(I)-1                                                BOUND2
      I1=I                                                           BOUND2
      KY=0                                                           BOUND2
      GO TO 8                                                        BOUND2
   7  IC(I-1)=IC(I)                                                  BOUND2
      I1=I-1                                                         BOUND2
   8  K=IC(I)                                                        BOUND2
      ID(I-1)=ID(I)-JX-KY                                            BOUND2
      DO 10 J=K,I1                                                   BOUND2
      IF(J .EQ. IA) JX=JX+1                                          BOUND2
      KXX=KX+JX                                                      BOUND2
      A(KX)=A(KXX)                                                   BOUND2
  10  KX=KX+1                                                        BOUND2
  20  NH=NH-1                                                        BOUND2
      IH=IH-1                                                        BOUND2
  30  CONTINUE                                                       BOUND2
      RETURN                                                         BOUND2
```

```
      END                                                              BOUND2
*DECK GAUSS
      SUBROUTINE  GAUSS(A,F,D,IC,ID,L,N,NN,NDCOMP)                     GAUSS
      DIMENSION A(1),IC(1), ID(1),F(NN,L),D(NN,L)                      GAUSS
      IF(NDCOMP .EQ. 1)GO TO 15                                        GAUSS
      DO 10 I=1,N                                                      GAUSS
      I1=I-1                                                           GAUSS
      DO  9 J=I,N                                                      GAUSS
      IF(IC(J) .GT. I)GO TO 9                                          GAUSS
      IX=ID(J)-J+I                                                     GAUSS
      IF(I1 .EQ. 0)GO TO 8                                             GAUSS
      DO  7 K=1,I1                                                     GAUSS
      IF(IC(J) .GT. K .OR. IC(I) .GT. K)GO TO 7                        GAUSS
      KX=ID(I)-I+K                                                     GAUSS
      KY=ID(J)-J+K                                                     GAUSS
      KZ=ID(K)                                                         GAUSS
      A(IX)=A(IX)-(A(KX)*A(KZ)* A(KY))                                 GAUSS
    7 CONTINUE                                                         GAUSS
    8 IF(I .EQ. J)GO TO 9                                              GAUSS
      KZ=ID(I)                                                         GAUSS
      IF(A(KZ) .EQ. 0.)GO TO 100                                       GAUSS
      A(IX)=A(IX)/A(KZ)                                                GAUSS
    9 CONTINUE                                                         GAUSS
   10 CONTINUE                                                         GAUSS
   15 DO 40 K=1,L                                                      GAUSS
      DO 30 I=1,N                                                      GAUSS
      D(I,K)=F(I,K)                                                    GAUSS
      I1=I-1                                                           GAUSS
      IF(I1 .EQ. 0) GO TO 30                                           GAUSS
      DO 20 J=1,I1                                                     GAUSS
      IF(IC(I) .GT. J)GO TO 20                                         GAUSS
      IX=ID(I)-I+J                                                     GAUSS
      D(I,K)=D(I,K)-A(IX)*D(J,K)                                       GAUSS
   20 CONTINUE                                                         GAUSS
   30 CONTINUE                                                         GAUSS
   40 CONTINUE                                                         GAUSS
      DO 70 I=1,N                                                      GAUSS
      KX=ID(I)                                                         GAUSS
      DO 70 K=1,L                                                      GAUSS
   70 D(I,K)=D(I,K)/A(KX)                                              GAUSS
      DO 90 K=1,L                                                      GAUSS
      IX=N                                                             GAUSS
      DO 90 I=2,N                                                      GAUSS
      IX=IX-1                                                          GAUSS
      I1=I-1                                                           GAUSS
      KX=IX                                                            GAUSS
      DO 80 J=1,I1                                                     GAUSS
      KX=KX+1                                                          GAUSS
      IF(IC(KX) .GT. IX)GO TO 80                                       GAUSS
      KY=ID(KX)-KX+IX                                                  GAUSS
      D(IX,K)=D(IX,K)-A(KY)*D(KX,K)                                    GAUSS
   80 CONTINUE                                                         GAUSS
   90 CONTINUE                                                         GAUSS
      GO TO 110                                                        GAUSS
  100 WRITE(6,120)                                                     GAUSS
  120 FORMAT(///2X,21HSTRUCTURE IS UNSTABLE///)                        GAUSS
  110 RETURN                                                           GAUSS
      END                                                              GAUSS
*DECK RESTOR
      SUBROUTINE RESTOR( D,IB,N,NB,L,NN)                               RESTOR
      DIMENSION D(NN,L),IB(1),TDR1(10),TDR2(10)                        RESTOR
```

```
      NH=N-NB                                                          RESTOR
      IH=1                                                             RESTOR
 1    I=IB(IH)                                                         RESTOR
      IF(I.GT.NH) GO TO 7                                              RESTOR
      DO 2 K = 1,L                                                     RESTOR
      TDR1(K) = D(I,K)                                                 RESTOR
    2 D(I,K) = 0.0                                                     RESTOR
    3 J = I + 1                                                        RESTOR
      IF (J .GT. NH) GO TO 5                                           RESTOR
      DO 4 K=1,L                                                       RESTOR
 4    TDR2(K)=D(J,K)                                                   RESTOR
 5    DO 6 K=1,L                                                       RESTOR
      D(J,K) =TDR1(K)                                                  RESTOR
 6    TDR1(K)=TDR2(K)                                                  RESTOR
      IF(I.GE.NH) GO TO 9                                              RESTOR
      I=I+1                                                            RESTOR
      GO TO 3                                                          RESTOR
 7    DO 8 K=1,L                                                       RESTOR
 8    D(I,K)=0.                                                        RESTOR
 9    IF(IH.GE.NB) GO TO 10                                            RESTOR
      IH=IH+1                                                          RESTOR
      NH=NH+1                                                          RESTOR
      GO TO 1                                                          RESTOR
10    CONTINUE                                                         RESTOR
      RETURN                                                           RESTOR
      END                                                              RESTOR
*DECK ELFORC
      SUBROUTINE ELFORC(AA,DR,EDR,MM,MA,MB,MC,MD,NNODES,LOADS,NN)      ELFORC
      DIMENSION AA(3,3),DR(NN,LOADS),EDR(12,LOADS), NCON(4)            ELFORC
      NCON(1)=MM*(MA    -1)+1                                          ELFORC
      NCON(2)=MM*(MB    -1)+1                                          ELFORC
      IF(NNODES     .GE. 3)NCON(3)=MM*(MC    -1)+1                   ELFORC
      IF(NNODES     .GE. 4)NCON(4)=MM*(MD    -1)+1                   ELFORC
      NND=NNODES                                                       ELFORC
      IF(NND .GT. 4)NND=4                                              ELFORC
      NDSP=1                                                           ELFORC
      IF(NND .GT. 2)NDSP=2                                             ELFORC
      DO 86 K=1,LOADS                                                  ELFORC
      KH=1                                                             ELFORC
      DO 86 KK=1,NND                                                   ELFORC
      DO 86 I=1,NDSP                                                   ELFORC
      KX=NCON (KK)                                                     ELFORC
      EDR(KH,K)=0                                                      ELFORC
      DO 85 J=1,MM                                                     ELFORC
      EDR(KH,K)=EDR(KH,K)+AA(I,J)*DR(KX,K)                             ELFORC
   85 KX=KX+1                                                          ELFORC
   86 KH=KH+1                                                          ELFORC
      RETURN                                                           ELFORC
      END                                                              ELFORC
*DECK QLSTRS
      SUBROUTINE QLSTRS(EDR,EDDR,XI,ETA,MAA,MBB,MCC,SX,SY,SXY,EFSTRS,
     1EXM,SNMAX,EE,AX,AY,AZ,ALS,LOADS,SSX,SSY,SSXY,EFFSTR,KTR,EKK,ENG,
     2NND)
      DIMENSION EDR(12,LOADS),EDDR(12,LOADS),XI(1),ETA(1),MAA(1),MBB(1), QLSTRS
     1MCC(1),SX(1),SY(1),SXY(1),EFSTRS(1),SSX(4,LOADS),SSY(4,LOADS),    QLSTRS
     2SSXY(4,LOADS),EFFSTR(4,LOADS),KTR(1),EKK(12,12),ENG(1),ENGG(10)   QLSTRS
     3,ALS(5),EE(3,3),EXM(1),SNMAX(1)
      DO 115 K=1,LOADS                                                 QLSTRS
      ENG(K)=0.                                                        QLSTRS
      EXM(K) = 0.0
      KX=0                                                             QLSTRS
```

```
      DO 115 I=9,10                                                    QLSTRS
      KX=KX+1                                                          QLSTRS
      EDR(I,K)=0.                                                      QLSTRS
      DO 114 J=1,8                                                     QLSTRS
  114 EDR(I,K)=EDR(I,K)+EKK(KX,J)*EDR(J,K)                             QLSTRS
      EDR(I,K)=-EDR(I,K)                                               QLSTRS
  115 CONTINUE                                                         QLSTRS
      DO 116 K=1,LOADS                                                 QLSTRS
      SNMAX(K) = 0.0
      EDDR(5,K)=EDR(9,K)                                               QLSTRS
  116 EDDR(6,K)=EDR(10,K)                                              QLSTRS
      KX=1                                                             QLSTRS
      KY=3                                                             QLSTRS
      DO 200 I=1,4                                                     QLSTRS
      IF(I .LT. 4)GO TO 117                                            QLSTRS
      KX=1                                                             QLSTRS
      KY=7                                                             QLSTRS
  117 DO 119 J=1,2                                                     QLSTRS
      DO 118 K=1,LOADS                                                 QLSTRS
      EDDR(J,K)=EDR(KX,K)                                              QLSTRS
  118 EDDR(J+2,K)=EDR(KY,K)                                            QLSTRS
      KX=KX+1                                                          QLSTRS
  119 KY=KY+1                                                          QLSTRS
      CALL STRESS(EDDR,XI,ETA,MAA(I),MBB(I),MCC(I),SX,SY,SXY,EFSTRS,   QLSTRS
     1EXM,EE,AX,AY,AZ,ALS, LOADS,ENGG,TRIANG,NND)
      DO 201 J=1,LOADS                                                 QLSTRS
      IF (ABS(SNMAX(J)) .LT. ABS(EXM(J)))SNMAX(J) = EXM(J)
      ENG(J)=ENG(J)+ENGG(J)                                            QLSTRS
      SSX(I,J)=SX(J)                                                   QLSTRS
      SSY(I,J)=SY(J)                                                   QLSTRS
      SSXY(I,J)=SXY(J)                                                 QLSTRS
      EFFSTR(I,J)=EFSTRS(J)                                            QLSTRS
      IF(NND .GT. 4)EFFSTR(I,J)=ABS(SXY(J)/ALS(5))                     QLSTRS
  201 CONTINUE                                                         QLSTRS
  200 CONTINUE                                                         QLSTRS
      DO 205 J=1,LOADS                                                 QLSTRS
      AMAX=0.                                                          QLSTRS
      DO 204 I=1,4                                                     QLSTRS
      IF(AMAX .GT. EFFSTR(I,J))GO TO 204                               QLSTRS
      AMAX=EFFSTR(I,J)                                                 QLSTRS
      KTR(J)=I                                                         QLSTRS
  204 CONTINUE                                                         QLSTRS
  205 CONTINUE                                                         QLSTRS
      RETURN                                                           QLSTRS
      END                                                              QLSTRS
*DECK STRESS
      SUBROUTINE STRESS(UV,X,Y,MA,MB,MC,SX,SY,SXY,EFST,EXM,EE,AX,AY,AZ,
     1ALS, L,ENG, TRIANG,NND)
      DIMENSION UV(12,L),X(1),Y(1),SX(1),SY(1),SXY(1),EX(10),EY(10),   STRESS
     1EXY(10),A(3,3),EXM(1),  EFST(1),ENG(1),EE(3,3),ALS(5)
      CALL CRAMER(A,TRIANG,X,Y,MA,MB,MC)                               STRESS
      DO 30 K=1,L                                                      STRESS
      EX(K)=0.                                                         STRESS
      EY(K)=0.                                                         STRESS
      EXY(K)=0.                                                        STRESS
      KX=0                                                             STRESS
      DO 20 I=1,3                                                      STRESS
      IX=I +KX                                                         STRESS
      EX(K)=EX(K)+A(1,I)*UV(IX,K)                                      STRESS
      EY(K)=EY(K)+A(2,I)*UV(IX+1,K)                                    STRESS
      EXY(K)=EXY(K)+A(2,I)*UV(IX,K)+A(1,I)*UV(IX+1,K)                  STRESS
```

```
   20  KX=KX+1                                                          STRESS
       EXM(K) = EX(K)*AX + EY(K)*AY + EXY(K)*AZ
   30  CONTINUE                                                         STRESS
       DO 40 K=1,L                                                      STRESS
       SX(K)=EE(1,1)*EX(K)+EE(1,2)*EY(K)+EE(1,3)*EXY(K)                 STRESS
       SY(K)=EE(2,1)*EX(K)+EE(2,2)*EY(K)+EE(2,3)*EXY(K)                 STRESS
   40  SXY(K)=EE(3,1)*EX(K)+EE(3,2)*EY(K)+EE(3,3)*EXY(K)                STRESS
       DO 50 K=1,L                                                      STRESS
       ENG(K)=(SX(K)*EX(K)+SY(K)*EY(K)+SXY(K)*EXY(K))*TRIANG            STRESS
       IF(NND .GT. 4)ENG(K)=(SXY(K)*EXY(K))*TRIANG                      STRESS
       EX(K)=SX(K)*AX+SY(K)*AY+SXY(K)*2.*AZ                             STRESS
       EY(K)=SX(K)*AY+SY(K)*AX-SXY(K)*2.*AZ                             STRESS
   50  EXY(K)=-SX(K)*AZ+SY(K)*AZ+SXY(K)*(AX-AY)                         STRESS
       DO 90 K=1,L                                                      STRESS
       AAX=ALS(1)                                                       STRESS
       AAY=ALS(3)                                                       STRESS
       AAXY=ALS(5)                                                      STRESS
       IF(EX(K) .LT. 0.)AAX=ALS(2)                                      STRESS
       IF(EY(K) .LT. 0.)AAY=ALS(4)                                      STRESS
       EFST(K)=SQRT((EX(K)/AAX)**2+(EY(K)/AAY)**2-((EX(K)*EY(K))/       STRESS
      1(AAX*AAY))+(EXY(K)/AAXY)**2)                                     STRESS
   90  CONTINUE                                                         STRESS
       RETURN                                                           STRESS
       END                                                              STRESS
*DECK TRECON
       SUBROUTINE TRECON(EE,AA,XAG,YAG,ZAG,AX,AY,AZ,IANG,IND)           TRECON
       DIMENSION EE(3,3),AA(3,3),AE(3,3)                                TRECON
       CTA=COS(XAG)                                                     TRECON
       STA=SIN(XAG)                                                     TRECON
       IF(IANG .EQ. 1)GO TO 20                                          TRECON
       AX=COS(XAG)                                                      TRECON
       AY=COS(YAG)                                                      TRECON
       AZ=COS(ZAG)                                                      TRECON
       AXX=AX*AA(1,1)+AY*AA(1,2)+AZ*AA(1,3)                             TRECON
       AYY=AX*AA(2,1)+AY*AA(2,2)+AZ*AA(2,3)                             TRECON
       AX=SQRT(AXX**2+AYY**2)                                           TRECON
       CTA=AXX/AX                                                       TRECON
       STA=AYY/AX                                                       TRECON
   20  CONTINUE                                                         TRECON
       IF(IND    .EQ. 1)GO TO 25                                        TRECON
       SAVE=CTA                                                         TRECON
       PI4=COS(3.141592654 /4.)                                         TRECON
       IF(IND-3)26,27,28                                                TRECON
   26  CTA=-STA                                                         TRECON
       STA=SAVE                                                         TRECON
       GO TO 25                                                         TRECON
   27  CTA=PI4*(CTA-STA)                                                TRECON
       STA=PI4*(SAVE+STA)                                               TRECON
       GO TO 25                                                         TRECON
   28  CTA=-PI4*(CTA+STA)                                               TRECON
       STA=PI4*(SAVE-STA)                                               TRECON
   25  CONTINUE                                                         TRECON
       AX=CTA**2                                                        TRECON
       AY=STA**2                                                        TRECON
       AZ=CTA*STA                                                       TRECON
       DO 30 I=1,3                                                      TRECON
       AE(I,1)=EE(I,1)*AX+EE(I,2)*AY-EE(I,3)*2.*AZ                      TRECON
       AE(I,2)=EE(I,1)*AY+EE(I,2)*AX+EE(I,3)*2.*AZ                      TRECON
   30  AE(I,3)=EE(I,1)*AZ-EE(I,2)*AZ+EE(I,3)*(AX-AY)                    TRECON
       DO 40 I=1,3                                                      TRECON
       EE(1,I)=AX*AE(1,I)+AY*AE(2,I)-2.*AZ*AE(3,I)                      TRECON
```

```
      EE(2,I)=AY*AE(1,I)+AX*AE(2,I)+2.*AZ*AE(3,I)                    TRECON
   40 EE(3,I)=AZ*AE(1,I)-AZ*AE(2,I)+(AX-AY)*AE(3,I)                  TRECON
      RETURN                                                         TRECON
      END                                                            TRECON
*DECK ELSTIC
      SUBROUTINE ELSTIC(E1,E2,PMU,SM,EE)                             ELSTIC
      DIMENSION EE(3,3)                                              ELSTIC
      PMU1=1.-(PMU**2)*(E2/E1)                                       ELSTIC
      EE(1,1)=E1/PMU1                                                ELSTIC
      EE(2,1)=(E2*PMU)/PMU1                                          ELSTIC
      EE(3,1)=0.                                                     ELSTIC
      EE(2,2)=E2/PMU1                                                ELSTIC
      EE(3,2)=0.                                                     ELSTIC
      EE(3,3)=SM                                                     ELSTIC
      DO 18 I=1,2                                                    ELSTIC
      IP=I+1                                                         ELSTIC
      DO 18 J=IP,3                                                   ELSTIC
   18 EE(I,J)=EE(J,I)                                                ELSTIC
      RETURN                                                         ELSTIC
      END                                                            ELSTIC
*DECK PRNTDR
      SUBROUTINE PRNTDR(A,B,X,Y,Z,N,M,L,NJ,NP,NN)                    PRNTDR
      DIMENSION A(NN,L),B(NN,L),X(1),Y(1),Z(1)                       PRNTDR
      NP=NP+1                                                        PRNTDR
      LINES=1                                                        PRNTDR
      WRITE(6,1)NP                                                   PRNTDR
      WRITE(6,2)                                                     PRNTDR
      DO 10 I=1,NJ                                                   PRNTDR
      IF (LINES+L-54)4,3,3                                           PRNTDR
    3 LINES=1                                                        PRNTDR
      WRITE(6, 1) NP                                                 PRNTDR
      WRITE(6, 2)                                                    PRNTDR
      NP=NP+1                                                        PRNTDR
    4 KH=M*I                                                         PRNTDR
      KHH=KH-M+1                                                     PRNTDR
      IF(M .LT. 3)GO TO 11                                           PRNTDR
      WRITE(6, 9)I,X(I),Y(I),Z(I),( A(J,1),J=KHH,KH),( B(J,1),J=KHH,KH) PRNTDR
      GO TO 12                                                       PRNTDR
   11 WRITE(6, 5)I,X(I),Y(I), ( A(J,1),J=KHH,KH),( B(J,1),J=KHH,KH)  PRNTDR
   12 IF(L .EQ. 1) GOTO 8                                            PRNTDR
      DO 7   K=2,L                                                   PRNTDR
      IF(M .LT. 3)GO TO 13                                           PRNTDR
      WRITE (6,  6) ( A(J,K) ,J=KHH,KH), ( B(J,K), J=KHH,KH)         PRNTDR
      GO TO 7                                                        PRNTDR
   13 WRITE (6, 15) ( A(J,K) ,J=KHH,KH), ( B(J,K), J=KHH,KH)         PRNTDR
    7 CONTINUE                                                       PRNTDR
    8 LINES =LINES +L+1                                              PRNTDR
      IF(L      .EQ. 1)LINES=LINES-1                                 PRNTDR
   10 CONTINUE                                                       PRNTDR
    1 FORMAT(1H1,120X,5HPAGE ,I3/)                                   PRNTDR
    2 FORMAT(  1X,5HJOINT,8X,2H-X,8X,2H-Y,8X,2H-Z,8X,7HFORCE-X,      PRNTDR
     17X,7HFORCE-Y,7X,7HFORCE-Z,8X,7HDISPL-X,10X,7HDISPL-Y,10X,      PRNTDR
     27HDISPL-Z//)                                                   PRNTDR
    9 FORMAT(/I5,F14.3,F10.3,F10.3,F12.3,F14.3,F14.3,1PE18.8,        PRNTDR
     11PE17.8,1PE17.8)                                               PRNTDR
    5 FORMAT(/I5,F14.3,F10.3,10X,F12.3,F14.3,14X,1PE18.8,1PE17.8)    PRNTDR
    6 FORMAT(39X,F12.3,F14.3,F14.3,1PE18.8,1PE17.8,1PE17.8)          PRNTDR
   15 FORMAT(39X,F12.3,F14.3,14X,1PE18.8,1PE17.8)                    PRNTDR
      RETURN                                                         PRNTDR
      END                                                            PRNTDR
*DECK LAYCALC
```

```
      SUBROUTINE LAYCALC (L,AAE,LAM,TFFR1,TFFR2,NZDEG,NNDEG,          LAYCALC
     1NFDEG,THKLAM,LFLAG1,LFLAG2,NKIND,NCOUNT)                        LAYCALC
      DIMENSION NZDEG(1),NNDEG(1),NFDEG(1),LFLAG1(1),LFLAG2(1)        LAYCALC
     1,NKIND(1)                                                       LAYCALC
      NCOUNT = NCOUNT + 1                                             LAYCALC
      I = NCOUNT                                                      LAYCALC
      NNDEG(L) = 0                                                    LAYCALC
      NZDEG(L) = 0                                                    LAYCALC
      NFDEG(L) = 0                                                    LAYCALC
      LFLAG1(L) = 0                                                   LAYCALC
      LFLAG2(L) = 0                                                   LAYCALC
      NKIND(I) = L                                                    LAYCALC
C     90 DEG FIBER DRECTION                                           LAYCALC
      A = TFFR2/THKLAM                                                LAYCALC
      LA = A                                                          LAYCALC
      IF (LA .GT. 0) GO TO 10                                         LAYCALC
      NNDEG(L) = 1                                                    LAYCALC
      GO TO 50                                                        LAYCALC
   10 IF ((A-LA) .GT. .5) GO TO 15                                    LAYCALC
      NNDEG(L) = LA                                                   LAYCALC
      GO TO 50                                                        LAYCALC
   15 NNDEG(L) = LA + 1                                               LAYCALC
   50 CONTINUE                                                        LAYCALC
C     0 DEG FIBER DIRECTION                                           LAYCALC
      B = TFFR1/THKLAM                                                LAYCALC
      LB = B                                                          LAYCALC
      IF (LB .GT. 0) GO TO 60                                         LAYCALC
      NZDEG(L) = 1                                                    LAYCALC
      GO TO 100                                                       LAYCALC
   60 IF ((B-LB) .GT. .5) GO TO 65                                    LAYCALC
      NZDEG(L) = LB                                                   LAYCALC
      GO TO 100                                                       LAYCALC
   65 NZDEG(L) = LB + 1                                               LAYCALC
  100 CONTINUE                                                        LAYCALC
C     45 DEG FIBER DIRECTION                                          LAYCALC
      C = (AAE - TFFR1 - TFFR2)/THKLAM                                LAYCALC
      LC = C                                                          LAYCALC
      K = MOD(LC,2)                                                   LAYCALC
      IF (K .NE. 0) GO TO 110                                         LAYCALC
      NFDEG(L) = LC                                                   LAYCALC
      GO TO 150                                                       LAYCALC
  110 IF (LC .GT. 1) GO TO 160                                        LAYCALC
      NFDEG(L) = 2                                                    LAYCALC
      GO TO 150                                                       LAYCALC
  160 NFDEG(L) = LC + 1                                               LAYCALC
  150 CONTINUE                                                        LAYCALC
C     CHECK                                                           LAYCALC
      LT = NNDEG(L) + NZDEG(L) + NFDEG(L)                             LAYCALC
      IF (LT .EQ. LAM) GO TO 1000                                     LAYCALC
      IF (LT .GT. LAM) GO TO 800                                      LAYCALC
      NZDEG(L) = NZDEG(L) + 1                                         LAYCALC
      LFLAG1(L) = 1                                                   LAYCALC
      LT = LT + 1                                                     LAYCALC
      IF (LT .EQ. LAM) GO TO 1000                                     LAYCALC
      LFLAG2(L) = 1                                                   LAYCALC
      GO TO 1000                                                      LAYCALC
  800 NZDEG(L) = NZDEG(L) - 1                                         LAYCALC
      LFLAG1(L) = 1                                                   LAYCALC
 1000 RETURN                                                          LAYCALC
      END                                                             LAYCALC
*DECK LAYPR
```

```
      SUBROUTINE LAYPR(LAM,NZDEG,NNDEG,NFDEG,LFLAG1,LFLAG2,                LAYPR
     1NKIND,NCOUNT,NFAC)                                                   LAYPR
      DIMENSION LAM(1),NZDEG(1),NNDEG(1),NFDEG(1),LFLAG1(1),LFLAG2(1),     LAYPR
     1NKIND(1)                                                             LAYPR
      WRITE(6,10)                                                          LAYPR
   10 FORMAT(1H1,30X,18HCOMPOSITE ELEMENTS///)                             LAYPR
      WRITE(6,20)                                                          LAYPR
   20 FORMAT(5X,4HMEMB,5X,9HTOTAL NO.,5X,52HTHE NUMBER OF LAYERS IN EACH   LAYPR
     1 OF THE FIBER DIRECTIONS/,14X,9HOF LAYERS,13X,1H0,15X,2H90,          LAYPR
     214X,2H45//)                                                          LAYPR
      DO 100 L = 1,NCOUNT                                                  LAYPR
      I = NKIND(L)                                                         LAYPR
      IF (LFLAG1(I) .EQ. 0) GO TO 50                                       LAYPR
      IF (LFLAG2(I) .EQ. 0) GO TO 25                                       LAYPR
C     OUTPUT FOR THIS LINE SHOULD BE NOTED BY THE USER                     LAYPR
      WRITE(6,30) I,LAM(I),NZDEG(I),NNDEG(I),NFDEG(I)                      LAYPR
   30 FORMAT(6X,I3,8X,I3,15X,I3,14X,I3,13X,I3,20X,2H**)                    LAYPR
      GO TO 100                                                            LAYPR
   25 WRITE(6,32) I,LAM(I),NZDEG(I),NNDEG(I),NFDEG(I)                      LAYPR
   32 FORMAT(6X,I3,8X,I3,15X,I3,14X,I3,13X,I3,10X,1H*)                     LAYPR
      GO TO 100                                                            LAYPR
   50 WRITE(6,34) I,LAM(I),NZDEG(I),NNDEG(I),NFDEG(I)                      LAYPR
   34 FORMAT(6X,I3,8X,I3,15X,I3,14X,I3,13X,I3)                             LAYPR
  100 CONTINUE                                                             LAYPR
      NFAC = NCOUNT                                                        LAYPR
      NCOUNT = 0                                                           LAYPR
      RETURN                                                               LAYPR
      END                                                                  LAYPR
```

* U.S.Government Printing Office: 1989 - 648-056/04345